User's Manual for

Dataman-48Pro

Universal 48-pindriver Programmer with USB/LPT interface and ISP capability

Dataman-40Pro

Universal 40-pindriver Programmer with USB interface and ISP capability

July 2005

Information provided in this manual is intended to be accurate at the moment of release, but we continuously improve all our products. Please check for an updated manual on our website at **www.dataman.com**.

Dataman Programmers Ltd assumes no responsibility for misuse of this manual.

Dataman Programmers Ltd reserves the right to make changes or improvements to the product described in this manual at any time without notice. This manual contains names of companies, software products, etc., which may be trademarks of their respective owners. Dataman Programmers Ltd respects those trademarks.

ZLI-0294A

How to use this manual

Dear customer,

Thank you for purchasing a high quality Dataman programmer!

This manual explains how to install the control program and how to use your programmer. It is assumed that the user has some experience with computers and software installation . Once you have installed the control program, we recommend you consult the context sensitive HELP within the control program rather than the printed User's Manual. Revisions are implemented in the context sensitive help before the printed User's Manual.

Note: *This User's manual is used for two different programmers. Please read the section(s) relevant to your respective programmer.*

This manual contains two main sections:

Quick Start

Read this section if you are an experienced user. You will find only specific information regarding installation of the control program and use of your programmer. For more detailed instructions you may read the **Detailed description** section or the **Troubleshooting** chapter for your respective programmer.

Detailed description

Read this section if you are a less experienced user or if you need additional information. All programmer features are described in this section along with details regarding installation of the control program. Read this section to explore all of the features provided by your programmer.

---

*We continuously update our manual. You may find the latest version from our website (**www.dataman.com**).*

Table of contents

Conventions used in the manual

References to the control program functions are in bold, e.g. **Load**, **File**, **Device**, etc. References to control keys are written in brackets **<>**, e.g. **<F1>.**

Terminology used in the manual:

Device any kind of programmable integrated circuits or programmable devices

ZIF socket Zero Insertion Force socket used for insertion of target device

Buffer part of memory or disk, used for temporary data storage

Printer port type of port of PC (parallel), which is primarily dedicated for printer connection.

HEX data format - format of data file, which may be read with standard text viewers; e.g. byte 5AH is stored as characters '5' and 'A', which is ASCII bytes 35H and 41H. One line of this HEX file (one record) contains start address and data bytes. All records are secured with a checksum.

Introduction

This user's manual covers the following programmers: **Dataman-48Pro** and **Dataman-40Pro**.

Dataman-48Pro is a fast universal USB/LPT programmer and logic IC tester with 48 powerful pindrivers. Using a built-in in-circuit serial programming (ISP) connector, the programmer is able to program ISP capable chips. Also, the modular design allows new devices to be easily added to the device list. The Dataman-48Pro is a true universal and a low cost programmer, providing the most cost effective programmer in today's market.

Dataman-40Pro is a small, fast and powerful USB programmer with many supported programmable devices. Using a built-in, in-circuit serial programming (ISP) connector, this programmer is able to program ISP capable chips. This programmer is also designed to allow new devices to be easily added to the device list.

These programmers work with almost any IBM PC Pentium compatible or higher, portable or desktop personal computers. No special interface card is required to connect to the PC.

All programmers function flawlessly on systems running Windows 95/98/Me/NT/2000/XP.

These programmers are driven by an **easy-to-use, control program** with pull-down menus, hot keys and online help. The Control program is the same for both of these programmers: Dataman-48Pro and Dataman-40Pro.

Advanced design, including protection circuits, original brand components and careful manufacturing allows us to provide a **three-year warranty** on parts and labour for the programmers (limited 25,000 cycle warranty on ZIF socket). This warranty is valid for customers, who purchase a programmer directly from Dataman. The warranty conditions of Dataman distributors may differ depending the law system or reseller's warranty policy.

Free additional services:

- free technical support (phone/fax/e-mail).
- free lifetime software update via Web site.
- **AlgOR** (Algorithm On Request) service allows you to receive software support for programming devices not yet available in the current device list.

Free software updates are available from our Internet address **www.dataman.com.**

Products configuration

Before installing and using your programmer, please carefully check that your package includes all next mentioned parts.

	programmer	LPT cable	USB cable	power supply	diagnostic POD	ISP cable	ZIF anti-dust cover	User's manual	registration card	shipping case
Dataman-48Pro	•	•	•	•	•	•	•	•	•	•
Dataman-40Pro	•	-	•	•	•	•	•	•	•	•

If you find any discrepancy with respective parts list and/or if any of these items are damaged, please contact your distributor immediately.

PC requirements

Minimal PC requirements

- PC Pentium 166
- 32MB RAM
- one CD drive
- HDD, 40 MB free space
- operating system Windows 95/98/Me/NT/2000/XP
- USB port 1.1 or later
- one parallel (LPT) port for dedicated use (Dataman-48Pro parallel mode)

Recommended PC requirements

- Pentium PC III 800 MHz or higher
- 256 MB free RAM
- one CD drive
- HDD, 50 MB free space
- operating system: Windows XP
- LPT printer port supporting EPP/ECP modes (for programmers connected via LPT port)
- USB port ver. 1.1 or later (for programmers connected via USB port)

Note: *For convenience, we suggest that you use a supplementary multi I/O card to provide an additional printer port (LPT2 for example), in order to avoid sharing the same LPT port between printer and programmer.*

Quick Start

Installing programmer hardware

- switch off the PC and programmer
- connect the communication port of programmer to a printer port of the PC using the supplied cable
- switch on the PC
- connect the power supply adapter to the programmer

Installing the programmer software

Run the installation program from the CD (Setup.exe) and follow the on-screen instructions. Please, see our website for the latest information about the programmer hardware and software at **www.dataman.com**.

Using programmer software

Launch PG4UW.exe to enter the control program. The menu **Device** contains the device manipulation commands. The menu **File** contains commands for files and directories. The menu **Buffer** is to be used for buffer manipulation.

Programming a device - the shortest way

Use the hot key **<Alt+F5>** to input the device name and/or manufacturer to select the desired type of target device. If you want to copy an existing device, insert it into the ZIF socket of the programmer and then press key **<F7>**. If you want to program a target device with data from a disk press key **<F3>** and read the appropriate file into the buffer. Then insert your target device into the ZIF socket. To check if the device is blank - press key **<F6>.** Now you can program the device by pressing key **<F9>.** After programming you may perform additional verification by pressing key **<F8>.**

Detailed description

Dataman-48Pro

Introduction

Dataman-48Pro is the first member of a new USB-compatible generation of Windows 95/98/Me/NT/2000/XP based **universal programmers.** It is built to meet the demands of the development community for a fast, reliable, and versatile programmer.

Dataman-48Pro supports the silicon technologies of today and tomorrow for programmable devices without family-specific modules. Using the built-in in-circuit serial programming (ISP) connector, the programmer is able to program in-circuit.

Dataman-48Pro isn't only a programmer, but also a tester of TTL/CMOS logic ICs and memories. Furthermore, it allows generating user-definable test pattern sequences.

Dataman-48Pro provides a very competitive price and excellent hardware design for reliable programming. It is the most cost effective programmer in its class.

Dataman-48Pro provides **very fast programming** due to high-speed FPGA driven hardware and execution of time-critical routines inside the programmer.

Dataman-48Pro interfaces with the IBM PC Pentium compatible or higher, portable or desktop personal computers through USB (2.0) port or any standard parallel (printer) port. Programmer also supports IEEE1284 (ECP/EPP) high-speed parallel port. Support of USB/LPT port connection gives you choice to connect the Dataman-48Pro programmer to any PC, from latest notebook to an older desktop without USB port.

Dataman-48Pro has a FPGA based, totally reconfigurable, 48 powerful TTL pindrivers. They provide H/L/pull_up/pull_down and read capability for each pin on the socket. Advanced pindrivers incorporate **high-quality** and **high-speed** circuitry to deliver signals without overshoot or ground bounce for all supported devices. Pin drivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

Dataman-48Pro performs a device **insertion test** (wrong or backward position) and a **contact check** (poor contact pin-to-socket) before it programs each device. These capabilities, supported by **overcurrent protection** and **signature-byte check** help prevent chip damage due to operator error.

Built-in **protection circuits** eliminate damage of programmer and/or programmed device due environment or operator

failure. All the inputs of the Dataman-48Pro programmer, including the ZIF socket, connection to PC and power supply input, are **protected against ESD** up to 15kV.

Dataman-48Pro programmer performs device **verification** at the upper and lower limits of the supply voltage, which, improves programming yield, and guarantees long data retention.

Various **socket converters** are available to handle device in PLCC, SOIC, PSOP, SSOP, TSOP, TSSOP, TQFP, QFN (MLF), SDIP, BGA and other packages.

Dataman-48Pro programmer is driven by an **easy-to-use** control program with pull-down menus, hot keys and on-line help. Selecting a device can be done by class, manufacturer or by simply typing in part of the vendor name or part number.

Standard device-related commands (read, blank check, program, verify, erase) are enhanced by **test functions** (insertion test, signature-byte check), and **special functions** (autoincrement, production mode - start immediately after insertion of chip into socket).

All known data formats are supported. The file format is automatically detected during the load operation.

The rich-featured **autoincrement function** enables users to assign individual serial numbers to each programmed device - increments a serial number, or read the serial numbers or any programmed device identification signatures from a file.

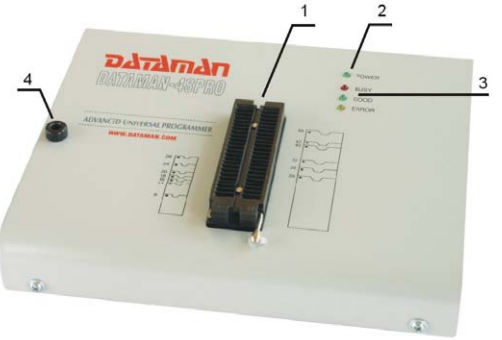
The software also provides information about the programmed device. The drawings of all available packages are provided. The software also provides explanation of chip labelling (the meaning of prefixes and suffixes at the chips) for each supported chip.

It is important to remember that in most cases new devices require **only a software update** due to the Dataman-48Pro universal programmer design. With our prompt service, new devices can be easily added. Please contact us for details.

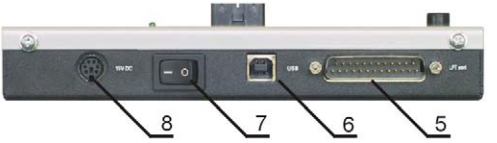
Advanced design including protection circuits, original brand components and careful manufacturing allows us to provide a **three-year warranty** on parts and labour for the Dataman-48Pro (limited 25,000-cycle warranty on ZIF socket).

Dataman-48Pro elements

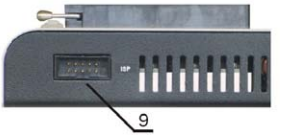
① 48 pin ZIF socket
② LED indicator power/sleep
③ LED indicators for work result
④ Jack for connecting ESD wrist strap



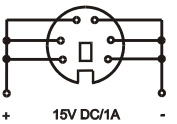
⑤ LPT connector for PC ↔ Dataman-48Pro communication cable
⑥ USB connector for PC ↔ Dataman-48Pro communication cable
⑦ Power switch
⑧ Power supply connector



⑨ ISP connector



Power supply connector



Connecting Dataman-48Pro to the PC

Using LPT port

Turn off your PC and programmer. Install the parallel cable included with your Dataman-48Pro programmer package to a free printer port on your PC. Connect the cable to the programmer and then to the PC. Make sure to tighten the connectors down with the thumb-screws to their respective ports. It may be uncomfortable to switch between a printer cable and programmer cable; however, it is not recommended to operate the Dataman-48Pro programmer through a printer switch. However, you can install a second multi-I/O controller in your computer, thus obtaining a supplementary printer port, such as LPT2. In this case, your printer may remain on LPT1 while the programmer is on LPT2.

Switch on the PC.
Plug in the power supply and then insert the mini-DIN connector into the programmer's connector labelled "15VDC". At this time all 'work result' LEDs (and 'POWER' LED) light up in succession and then switch off. Once the POWER LED is illuminated the Dataman-48Pro programmer is ready for use.
Next run the control program for Dataman-48Pro.

Caution! *If you don't want to switch off your PC when connecting the Dataman-48Pro, proceed as follows*:

- ***When connecting** the programmer to the PC: **FIRST** insert the **communications cable** and **THEN** the **power-supply connector.***
- ***When disconnecting** the programmer from the PC: **FIRST** disconnect the **power-supply connector** and **THEN** the **communication cable**.*

From the programmers' point of view the connecting and disconnecting sequence is irrelevant. Protection circuits on all programmer inputs keep it safe. **But for the safety of your PC we recommend the above sequence**.

Using the USB port

In this case, order of connecting USB cable and power supply to programmer is irrelevant.

Problems related to the Dataman-48Pro ⇔ PC connection

If you have any problems with Dataman-48Pro ⇔ PC connection, see section ***Common notes*** please.

Manipulation with the programmed device

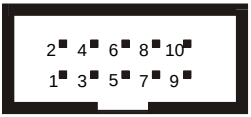
Select the device first, then insert it into the ZIF socket. Note that the ZIF socket is open with the lever up and closed with the lever down. The correct orientation of the device is shown on the picture near the ZIF socket on the programmer's cover. The programmed device can be removed from the socket when the BUSY LED turns off.

Note: *Programmer's protection electronics protect the target device and the programmer itself against either short or long-term power failures. However, it is not possible to guarantee the integrity of the target device due to incorrect, user-selected programming parameters. Do not remove the target device from the ZIF socket when the BUSY LED is on.*

In-system serial programming by Dataman-48Pro

For general definition, recommendation and direction about ISP see section ***Common notes / ISP*** please.

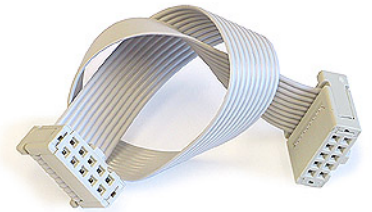
Description of Dataman-48Pro ISP connector



Front view of the ISP connector.

Depending on the device you want to program, the specification of ISP connector pins may change. You can find it in the control SW for the programmer (PG4UW), menu **Device / Device Info (Ctrl+F1)**. Please be aware that the ISP programming option of a given device must be selected. This is indicated by a “ISP” suffix after the name of selected device. For additional ISP information please refer to the device manufacturer.

Note: *Pin no. 1 is indicated by a triangle on ISP cable connector*.

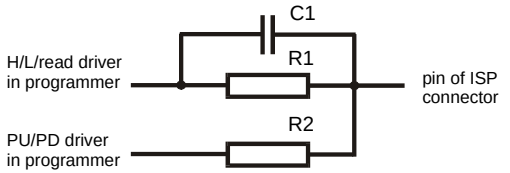


Dataman-48Pro ISP cable

Warnings:

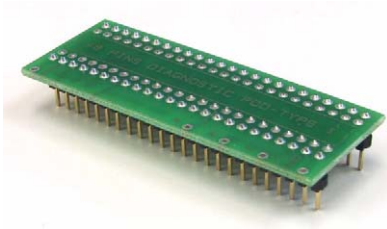
- ***When** you use Dataman-48Pro as **ISP programmer**, **do not insert** device into **ZIF socket**.*
- ***When** you program devices into **ZIF socket**, **do not insert** ISP cable to **ISP connector**.*
- *Use only **attached ISP cable**. When you use another ISP cable (other material, length…), programming may be unreliable.*
- ***Dataman-48Pro can supply** programming voltage (pin 1 of ISP connector) and target system voltage (pin 5 of ISP connector) with some limitations (see Technical specification / ISP connector), but target system **should not supply voltage to the Dataman-48Pro**.*
- *Dataman-48Pro applies programming voltage to the target device. If the programming voltage is different than expected, the Dataman-48Pro will abort the operation.*

Note: *H/L/read Dataman-48Pro driver*



Selftest and calibration

If you feel that your programmer does not work properly, please run the programmer selftest using the Diagnostic POD. The Diagnostic POD is included with the standard package. For optimal results, we recommend you run the programmer selftest and calibration every 6 months. See instructions for selftest in the **Diagnostics** menu of PG4UW.



Technical specification

HARDWARE

Base unit, DACs

- USB 2.0 port
- FPGA based IEEE 1284 slave printer port, up to 1MB/s transfer rate
- on-board intelligence: powerful microprocessor and FPGA based state machine
- three D/A converters for VCCP, VPP1, and VPP2, controllable rise and fall time
- VCCP range 0..8V/1A
- VPP1, VPP2 range 0..26V/1A
- autocalibration
- selftest capability
- protection against surge and ESD on power supply input, parallel port connection

Socket, pindriver

- 48-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 48-pin
- pindrivers: 48 universal
- VCCP / VPP1 / VPP2 can be connected to each pin
- ground for each pin
- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on all pindriver pins
- analog pindriver output level selectable from 1.8 V up to 26V
- current limitation, overcurrent shutdown, power failure shutdown
- ESD protection on each pin of socket (IEC1000-4-2: 15kV air, 8kV contact)
- continuity test: each pin is tested before every programming operation

ISP connector

- 10-pin male connector with one-way insertion lock
- 6 TTL pindrivers, provides H, L, CLK, pull-up, pull-down; level H selectable from 1.8V up to 5V to handle all (low-voltage including) devices.
- 1x VCCP voltage (range 2V..7V/100mA) and 1x VPP voltage (range 2V..25V/50mA)
- programmed chip voltage (VCCP) with both source/sink capability and voltage sense
- target system supply voltage (range 2V..6V/250mA)

DEVICE SUPPORT

Programmer, in ZIF socket

- EPROM: NMOS/CMOS, 2708*, 27xxx and 27Cxxx series, with 8/16 bit data width, full support for LV series
- EEPROM: NMOS/CMOS, 28xxx, 28Cxxx, 27EExxx series, with 8/16 bit data width
- Flash EPROM: 28Fxxx, 29Cxxx, 29Fxxx, 29BVxxx, 29LVxxx, 29Wxxx, 49Fxxx series, from 256Kbit to 32Mbit, with 8/16 bit data width, full support for LV series
- Serial E(E)PROM: 24Cxxx, 24Fxxx, 25Cxxx, 45Dxxx, 59Cxxx, 25Fxxx, 25Pxxx, 85xxx, 93Cxxx, NVM3060, MDAxxx series, full support for LV series
- Configuration (EE)PROM: XCFxxx, XC17xxxx, XC18Vxxx, EPCxxx, AT17xxx, 37LVxx
- 1-Wire E(E)PROM: DS1xxx, DS2xxx
- PROM: AMD, Harris, National, Philips/Signetics, Tesla, TI
- NV RAM: Dallas DSxxx, SGS/Inmos MKxxx, SIMTEK STKxxx, XICOR 2xxx, ZMD U63x series
- PLD: Altera: MAX 3000A, MAX 7000A, MAX 7000B, MAX 7000S, MAX7000AE
- PLD: Lattice: ispGAL22V10x, ispLSI1xxx, ispLSI1xxxEA, ispLSI2xxx, ispLSI2xxxA, ispLSI2xxxE, ispLSI2xxxV, ispLSI2xxxVE, ispLSI2xxxVL, LC4xxxB/C/V/ZC, M4-xx/xx, M4A3-xx/xx, M4A5-xx/xx, M4LV-xx/xx
- PLD: Xilinx: XC9500, XC9500XL, XC9500XV, CoolRunner XPLA3, CoolRunner-II
- other PLD: SPLD/CPLD series: AMI, Atmel, AMD-Vantis, Gould, Cypress, ICT, Lattice, NS, Philips, STM, VLSI, TI
- Microcontrollers 48 series: 87x41, 87x42, 87x48, 87x49, 87x50 series
- Microcontrollers 51 series: 87xx, 87Cxxx, 87LVxx, 89Cxxx, 89Sxxx, 89LVxxx, all manufacturers, Philips LPC series
- Microcontrollers Intel 196 series: 87C196 KB/KC/KD/KT/KR/...
- Microcontrollers Atmel AVR: AT90Sxxxx, ATtiny, ATmega series

- Microcontrollers Cypress: CY8Cxxxxx
- Microcontrollers ELAN: EM78Pxxx
- Microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17Cxxx, PIC18xxx, dsPIC series
- Microcontrollers Motorola: 68HC05, 68HC08, 68HC11 series
- Microcontrollers National: COP8xxx series
- Microcontrollers NEC: uPD78Pxxx series
- Microcontrollers Scenix (Ubicom): SXxxx series
- Microcontrollers SGS-Thomson: ST6xx, ST7xx, ST10xx series
- Microcontrollers TI: MSP430 and MSC121x series
- Microcontrollers ZILOG: Z86/Z89xxx and Z8xxx series
- Microcontrollers other: EM Microelectronic, Fujitsu, Goal Semiconductor, Hitachi, Holtek, Princeton, Macronix, Winbond, Infineon(Siemens), NEC, Samsung, Toshiba, ...

Programmer, through ISP connector

- Serial E(E)PROM: IIC series
- Microcontrollers Atmel: AT89Sxxx, AT90Sxxxx, ATtiny, ATmega series
- Microcontrollers Cypress: CY8C2xxxx
- Microcontrollers Elan: EM78Pxxx
- Microcontrollers EM Microelectronic: 4 and 8 bit series
- Microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17xxx, PIC18xxx, dsPIC series
- Microcontrollers Motorola/Freescale: HC08 GT, LJ, QY, QT series
- Microcontrollers Philips: LPC series
- Microcontrollers TI: MSP430
- PLD: Lattice: ispGAL22xV10x, ispLSI1xxxEA, ispLSI2xxxE, ispLSI2xxxV, ispLSI2xxxVE, ispLSI2xxxVL, M4-xx/xx, M4LV-xx/xx, M4A3-xx/xx, M4A5-xx/xx, LC4xxxB/C/V/ZC
- Various PLD (also by JAM player/JTAG support):
- Altera: MAX 3000A, MAX 7000A, MAX 7000B, MAX 7000S, MAX 9000, MAX II
- Xilinx: XC9500, XC9500XL, XC9500XV, CoolRunner XPLA3, CoolRunner-II

Notes:

- *Devices marked with '*' are obsolete and require an additional module for programming*
- *For all supported devices see actual **Device list** on our website **www.dataman.com***

I.C. Tester

- TTL type: 54,74 S/LS/ALS/H/HC/HCT series
- CMOS type: 4000, 4500 series
- static RAM: 6116.. 624000
- user definable test pattern generation

Package support

- package support includes DIP, PLCC, SOIC, PSOP, SSOP, TSOP, TSSOP, TQFP, QFN (MLF), SDIP, BGA and others
- support all devices in DIP with default socket
- support devices in non-DIP packages up to 48 pins with universal adapters
- programmer is compatible with third-party adapters for non-DIP support

Programming speed

Device	Operation	Time B
AT29C040A	programming and verify	21 sec
AM29DL323DB	programming and verify	38 sec
AM29DL640	programming and verify	76 sec
AT45D081	programming and verify	43 sec
AT89C51RD2	programming and verify	15 sec
PIC18F452	programming and verify	4 sec

System: *P4, 2,4GHz, USB 2.0, Windows XP*

SOFTWARE

- **Algorithms:** only manufacturer approved or certified algorithms are used. Custom algorithms are available at additional cost.
- **Algorithm updates:** software updates are available approx. every 2 weeks, free of charge.
- **Main features:** revision history, session logging, on-line help, device and algorithm information

Device operations

- **standard:**
  - intelligent device selection by device type, manufacturer or typed fragment of part name
  - automatic ID-based selection of EPROM/Flash EPROM
  - blank check, read, verify
  - program
  - erase
  - configuration and security bit program
  - illegal bit test
  - checksum
- **security**
  - insertion test, reverse insertion check
  - contact check
  - ID byte check
- **special**
  - production mode (automatic start immediately after device insertion)
  - auto device serial number increment

- statistic
- count-down mode

Buffer operations

- view/edit, find/replace
- fill/copy, move, byte swap, word/dword split
- checksum (byte, word)
- print

Supported file formats

- unformatted (raw) binary
- HEX: Intel, Intel EXT, Motorola S-record, MOS, Exormax, Tektronix, ASCII-space-HEX
- Altera POF, JEDEC (ver. 3.0.A), e.g. from ABEL, CUPL, PALASM, TANGO PLD, OrCAD PLD, PLD Designer ISDATA, etc.

PC system requirements

See section ***Introduction/ PC requirements***

GENERAL

- operating voltage 15..18V DC, max. 1A
- power consumption max. 12W active, about 2W inactive
- dimensions 160x190x42 mm (6.3x7.5x1.7 inch)
- weight (without external adapter) 900g (2lbs)
- temperature 5°C ÷ 40°C (41°F ÷ 104°F)
- humidity 20%..80%, non condensing

Package included

- Dataman-48Pro programmer
- connection cable PC-programmer, LPT port
- connection cable PC-programmer, USB port
- ISP cable
- diagnostic POD for selftest
- anti-dust cover for ZIF socket
- switching power adapter 100..240V AC/15V DC/1A
- user manual
- software
- registration card
- transport case

Additional services

- AlgOR
- free technical support (phone/fax/e-mail).
- free lifetime software update via Web site.

Dataman-40Pro

Introduction

Dataman-40Pro is the next member of the new generation of Windows 95/98/Me/NT/2000/XP based universal programmers. This programmer is built to meet the demands of development labs and field engineers for a fast, reliable, and versatile programming.

Dataman-40Pro is a small, fast and powerful programmer. Using the built-in, in-circuit serial programming (ISP) connector, the programmer is capable of in-circuit programming.

Dataman-40Pro is competitively priced, with excellent hardware design for reliable programming.

Dataman-40Pro offers very fast programming due to high-speed FPGA driven hardware and USB 2.0 full speed port. It is surely faster than competitors in this category.

Dataman-40Pro interfaces with the IBM PC Pentium compatible or higher, portable or desktop personal computers through USB port. Therefore you can take the programmer and move it to another PC without assembly/disassembly of the PC.

Dataman-40Pro has 40 powerful TTL pindrivers to provide H/L/pull_up/pull_down and read capability for each pin in the socket. Advanced pindrivers incorporate high-quality high-speed circuitry to deliver signals without overshoot or ground bounce for all supported devices. Pin drivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

The programmer performs a **device insertion test** (wrong device position in socket) and contact check (poor contact pin-to-socket) before it programs each device. These capabilities, supported by signature-byte check, help prevent chip damage due to operator error.

The programmer's hardware offers enough resources for a detailed **selftest.** The control program is able to check pindrivers, voltages levels, and communication between the programmer and the PC.

Dataman-40Pro programmer performs a device **verification** at the upper and lower limits of the supply voltage, which, improves programming yield, and guarantees long data retention.

Dataman-40Pro programmer is driven by an **easy-to-use** control program with pull-down menu, hot keys and on-line help. Selecting a device is performed by its class, by manufacturer or simply by typing a fragment of vendor name and/or part number.

Standard device-related commands (read, blank check, program, verify, erase) are enhanced by **test functions** (insertion test, signature-byte check), and **special functions** (autoincrement).

All known data formats are supported. File formats are automatically detected and converted during load.

The rich-featured **autoincrement function** enables to assign individual serial numbers to each programmed device - or simply increments a serial number, or the function enables to read serial numbers or any programmed device identification signatures from a file.

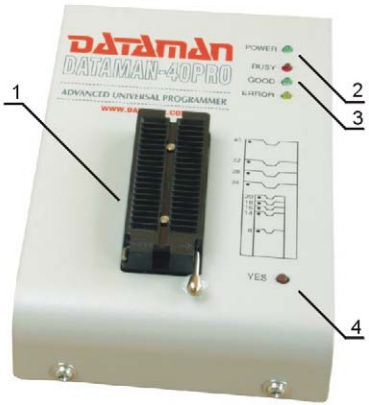
The software also provides information about the programmed device. The drawings of all available packages are provided. The software provides explanation of chip part numbers (the meaning of prefixes and suffixes of the chips) for each supported chip.

Various **socket converters** are available to handle device in PLCC, SOIC, SSOP, TSOP, TSSOP, TQFP, QFN (MLF) and other packages.

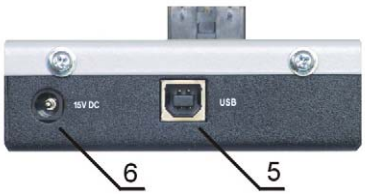
Advanced design, including protection circuits, original brand components and careful manufacturing allows us to provide a **three-year warranty** on parts and labour for the Dataman-40Pro (limited 25,000-cycle warranty on ZIF socket).

Dataman-40Pro elements

① 40 pin ZIF socket
② LED power/sleep
③ LED, which indicate work result
④ YES! button



⑤ USB connector for PC ↔ Dataman-40Pro communication cable
⑥ Power supply connector



⑦ Connector for ISP

Power supply connector

Note: *Due to the low power consumption when in an inactive state, the Dataman-40Pro doesn't require power switch. When the power LED indicator glows with a low intensity the Dataman-40Pro is in inactive mode.*

Connecting Dataman-40Pro to PC

For Dataman-40Pro the order of connecting the USB cable and power supply to programmer is irrelevant.

Problems related to the Dataman-40Pro ⇔ PC connection

If you have any problems with Dataman-40Pro ⇔ PC connection, see section ***Common notes*** please.

Manipulation with the programmed device

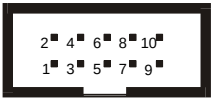
After selecting the desired part, you can insert it into the open ZIF socket (the lever is up) and then close socket (the lever is down) to begin programming. The correct orientation of the programmed device in ZIF socket is shown on the picture near ZIF socket on the programmer's cover. It is necessary to insert the device into the socket and to also close the socket. Make sure not to remove the device from the socket when the LED BUSY is on.

Warning: *The Dataman-40Pro programmer doesn't protect devices against critical situations, for example power failures and PC failure (interrupted cable...). Moreover, a device is usually destroyed in the programming mode due to forced interruption of the control program (Reset or switching the computer off) due to removing the connecting cable, or unplugging the programmed device from the ZIF socket. An incorrectly placed device in the ZIF socket can also cause damage.*

In-system serial programming by Dataman-40Pro

For general information, recommendations and directions about ISP see section ***Common notes / ISP*** please.

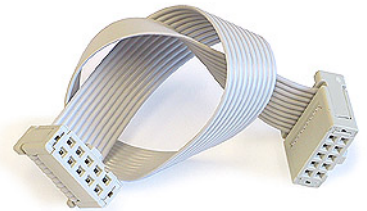
Description of Dataman-40Pro ISP connector



Front view of the ISP connector.

Depending on the device you want to program, the specification of ISP connector pins may change. You can find it in the control SW for the programmer (PG4UW), menu **Device / Device Info (Ctrl+F1)**. Please be aware that the ISP programming option of a given device must be selected. This is indicated by a "ISP" suffix after the name of selected device. For additional ISP information please refer to the device manufacturer.

Note: *Pin no. 1 is indicated by a triangle on ISP cable connector*..

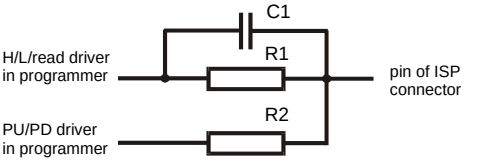


Dataman-40Pro ISP cable

Warnings:

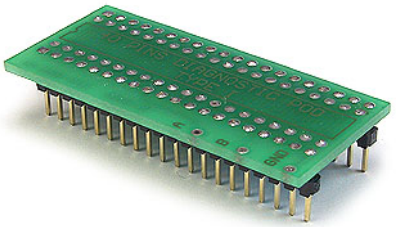
- ***When** you use Dataman-40Pro as a **ISP programmer**, **do not insert** a device in the **ZIF socket**.*
- ***When** you program devices in **ZIF socket**, **do not insert** the ISP cable in the **ISP connector**.*
- *Use only **included ISP cable**. When you use other ISP cables (other material, length…), programming may become unreliable.*
- ***Dataman-40Pro can only supply** programming voltage. Refer to ISP specs by device manufacturer for additional information.*
- *Dataman-40Pro apply programming voltage to target device and checks his value (target system can modify programming voltage). If the programming voltage is different as expected, no action with target device will be executed.*

Note: *H/L/read Dataman-40Pro driver*



Selftest and calibration

If you feel that your programmer does not work properly, please run the programmer selftest using the **Diagnostic POD**. The Diagnostic POD is included with the standard package. For optimal results, we recommend you run the programmer selftest and calibration every 6 months. See instructions for selftest in the **Diagnostics** menu of PG4UW.



Technical specification

HARDWARE

Programmer

- two D/A converters for VCCP and VPP, controllable rise and fall time
- VCCP range 0..7V/350mA
- VPP range 0..25V/200mA
- USB 2.0/1.1 compatible interface
- autocalibration
- selftest capability

ZIF socket, pindriver

- 40-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 40-pins
- pindriver: 40 TTL pindrivers, universal GND/VCC/VPP pindriver

- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on all pindriver pins, level H selectable from 1.8 V up to 5V
- in-circuit serial programming (ISP) capability included
- continuity test: each pin is tested before every programming operation

ISP connector

- 10-pin male type with missinsertion lock
- 6 TTL pindrivers, provides H, L, CLK, pull-up, pull-down; level H selectable from 1.8V up to 5V to handle all (low-voltage including) devices.
- 1x VCCP voltage (range 2V..7V/100mA) and 1x VPP voltage (range 2V..25V/50mA)
- programmed chip voltage (VCCP) with both source/sink capability and voltage sense

DEVICE SUPPORT

Programmer, in ZIF socket

- EPROM: NMOS/CMOS, 27xxx and 27Cxxx series, with 8/16 bit data width, full support of LV series (*1*2)
- EEPROM: NMOS/CMOS, 28xxx, 28Cxxx, 27EExxx series, with 8/16 bit data width, full support of LV series (*1*2)
- Flash EPROM: 28Fxxx, 29Cxxx, 29Fxxx, 29BVxxx, 29LVxxx, 29Wxxx, 49Fxxx series, with 8/16 bit data width, full support of LV series (*1*2)
- Serial E(E)PROM: 24Cxxx, 24Fxxx, 25Cxxx, 45Dxxx, 59Cxxx, 25Fxxx, 25Pxxx, 85xxx, 93Cxxx, full support for LV series (*1)
- Configuration (EE)PROM: XCFxxx, 37LVxx, XC17xxxx, EPCxxx, AT17xxx, LV series including
- NV RAM: Dallas DSxxx, SGS/Inmos MKxxx, SIMTEK STKxxx, XICOR 2xxx, ZMD U63x series
- PLD: series: Atmel, AMD-Vantis, Cypress, ICT, Lattice, NS, ... (*1)
- microcontrollers 51 series: 87Cxxx, 87LVxx, 89Cxxx, 89Sxxx, 89LVxxx, LPC series from Atmel, Atmel W&M, Intel, Philips, SST, Winbond (*1*2)
- microcontrollers Atmel AVR: ATtiny, AT90Sxxx, ATmega series (*1*2)
- Microcontrollers Cypress: CY8Cxxxxx
- Microcontrollers ELAN: EM78Pxxx
- Microcontrollers EM Microelectronic: 4 and 8 bit series
- microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17Cxxx, PIC18xxx, dsPIC series, 8-40 pins (*1*2)
- microcontrollers Scenix (Ubicom): SXxxx series

Programmer, through ISP connector

- Serial E(E)PROM: IIC series
- Microcontrollers Atmel: AT89Sxxx, AT90Sxxxx, ATtiny, ATmega series
- Microcontrollers Cypress: CY8C2xxxx
- Microcontrollers Elan: EM78Pxxx
- Microcontrollers EM Microelectronic: 4 and 8 bit series
- Microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17xxx, PIC18xxx, dsPIC series
- Microcontrollers Philips: LPC series

Notes:

- *(*1) - suitable adapters are available for non-DIL packages*
- *(*2) - there are only a few adapters for devices with more than 40 pins. If you need to program devices with more than 40 pins consider a more powerful programmer such as the Dataman-48Pro.*
- *For all supported devices see our **Device list** at **www.dataman.com**.*

I.C. Tester

- Static RAM: 6116 .. 624000

Programming speed

Device	Operation	Mode	Time
27C010	programming and verify	in ZIF	29 sec
AT29C040A	programming and verify	in ZIF	41 sec
AM29F040	programming and verify	in ZIF	95 sec
PIC16C67	programming and verify	in ZIF	10 sec
PIC18F452	programming and verify	in ZIF	7 sec
AT89C52	programming and verify	in ZIF	17 sec
PIC16F876A	programming and verify	ISP	5 sec
PIC12C508	programming and verify	ISP	3 sec

System: *P4, 2.4GHz, USB 2.0, Windows XP*

SOFTWARE

- **Algorithms:** only manufacturer approved or certified algorithms are used. Custom algorithms are available at additional cost.
- **Algorithm updates:** software updates are available approx. every 2 weeks, free of charge.
- **Main features:** revision history, session logging, on-line help, device and algorithm information

Device operations

- **standard:**
  - intelligent device selection by device type, manufacturer or partial part name
  - blank check, read, verify
  - program

  - erase
  - configuration and security bit program
  - illegal bit test
  - checksum
- **security**
  - insertion test
  - contact check
  - ID byte check
- **special**
  - auto device serial number increment
  - statistics
  - count-down mode

Buffer operations

- view/edit, find/replace
- fill, copy, move, byte swap, word/dword split
- checksum (byte, word)
- print

File load/save

- no download time because programmer is PC controlled
- automatic file type identification

Supported file formats

- unformatted (raw) binary
- HEX: Intel, Intel EXT, Motorola S-record, MOS, Exormax, Tektronix, ASCII-SPACE-HEX
- JEDEC (ver. 3.0.A), for example from ABEL, CUPL, PALASM, TANGO PLD, OrCAD PLD, PLD Designer ISDATA etc.

PC system requirements

See section ***Introduction/ PC requirements***

GENERAL

- operating voltage 15..20V DC, max. 500mA
- power consumption max. 6W active, 1.4W inactive
- dimensions 160x97x35 mm (6.3x3.8x1.4 inch)
- weight (without external power adapter) ca. 500g (17.65 oz)
- temperature 5°C ÷ 40°C (41°F ÷ 104°F)
- humidity 20%..80%, non condensing

Package included

- Dataman-40Pro programmer
- USB connection cable
- ISP cable

- diagnostic POD for selftest
- anti-dust cover for ZIF socket
- suitable power supply adapter
- user manual
- software
- registration card
- transport case

Additional services

- AlgOR
- free technical support
- free life-time software update via our website www.dataman.com

Software

The programmer software

The programmer package contains a CD with the control program, useful utilities and additional information. Permission is granted to freely copy the CD in order to demonstrate how the programmer works. Updates to this manual may be found at our website **www.dataman.com** .

Installing the programmer software

Installing the programmer software is very easy. Insert the CD to your CD-ROM drive and the install program should start automatically. The install program (setup.exe), will guide you through the installation process prior to using your programmer.

Program PG4UW.exe is the common control program between the Dataman-40Pro and Dataman-48Pro. This program will work on systems running Windows 95/98/Me/NT/2000/XP.

New versions of programmer software

In order to use all of the capabilities of the programmer, we recommend using the latest version of PG4UW. You may download the latest version of programmer software from our Internet site **www.dataman.com**. You may also obtain a CD with this file by postal mail (a mailing charge will apply). Please contact us for details.

Upgrading the programmer software

Copy PG4UWARC.exe to a temporary directory then launch it. After extraction you will see all available files needed for the installation process. Then redo a standard installation (run the Setup program). You may delete all files from the temporary folder after the installation process is complete.

Using the programmer software

The control program delivered by Dataman, included on the CD in your package, is granted to be free from any viruses at the moment of delivery.

Run the control program

In Windows environment: double click to icon PG4UW.

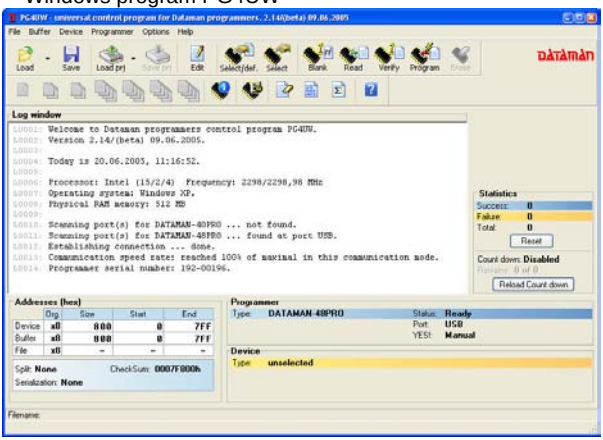
After starting the control program, PG4UW automatically scans all existing ports for a connected Dataman programmer. Program PG4UW will try to find all supported Dataman-48Pro and Dataman-40Pro programmers.

Note: *When PG4UW is started, it performs an integrity check, then displays a standard user menu and waits for your instructions.*

If the control program cannot communicate with the programmer, an error message appears on the screen, including error code and description of possible reasons (disconnected programmer, bad connection, power supply failure, incompatible printer port, ...). Eliminate the error source and press any key. If error condition still exists, the program resumes its operation in the demo mode and access to the programmer is not possible. If you cannot find the cause of the error, follow the instructions in **Troubleshooting** section. In addition, the control program checks communication with programmer prior to any operation with the programmed device.

Description of the user screen

Windows program PG4UW



Header bar	the name, copyright statement and version of the PG4UW control program
Menu bar	list of basic functions
Filename	information on the currently loaded file in buffer

Programmer window	information about the status of the programmer and PG4UW
Addresses window	organization, size, start and end addresses of the target device, buffer and file
Device window	all relevant information about the current target device
Help bar	a brief description of selected command

Menu selection is carried out in the normal GUI fashion - either by cursor moving plus pressing **<Enter>**, or by typing the highlighted letter in the wanted menu or - of course - by mouse. Hot-keys are available for even quicker selection of intensely used commands.

Note: *Data entered through keyboard is in HEX format, excepting ASCII blocks in Buffer/View/Edit command*.

List of hot keys

<F1>	Help	Calls Help
<F2>	Save	Save file
<F3>	Load	Load a file into the buffer
<F4>	Edit	Viewing/editing of buffer
<F5>	Select/default	Target-device selection from 10 last selected devices list
<Alt+F5>	Select/manual	Target-device selection by typing device/vendor name
<F6>	Blank	Blank check
<F7>	Read	Reads device's content into the buffer
<F8>	Verify	Compares contents of the target device with the buffer
<F9>	Program	Programs target device
<Alt+Q>	Exit without save	Terminates the PG4UW
<Alt+X>	Exit and save	Terminates the PG4UW and saving settings too
<Ctrl+F1>		Displays additional information about current device
<Ctrl+F2>	Erase	Fill's the buffer with a given value
<Ctrl+Shift+F2>		Fill's the buffer with random values.

File

This submenu is used for source files manipulation, settings and viewing directory, changes drives, changes start and finish address of buffer for loading and saving files by **binary**, **MOTOROLA**, **MOS Technology**, **Intel (extended) HEX**, **Tektronix, ASCII space**, **JEDEC**, and **POF** format. The menu

commands for loading and saving projects are located in this submenu too.

File / Load

Select the file format and load the data from specified file to the buffer. You can choose the format desired (**binary**, **MOTOROLA**, **MOS Technology**, **Tektronix**, **Intel (extended) HEX**, **ASCII space**, **JEDEC** and **POF**). The control program stores a last valid mask for file listing. You can save the mask into the config. file by command **Options / Save options**.

Selecting **Automatic file format recognition** tells program to detect file format automatically. When program can't detect file format from one of supported formats, the binary file format is assumed.

When the check box **Automatic file format recognition** is unchecked, the program allows user to manually select the file format from list of available formats on the panel **Selected file format**. Default set is from **Options / General options** in panel **Load file format** at tab **File options**.

Selecting **Buffer offset for loading** tells the program to set buffer offset for all data addresses, which will be written to buffer. This feature is useful for binary and all HEX formats. Using this one-shot setting disables current setting of native offset in menu **Options / General options** in panel **Negative offset for loading** at tab **Hex file options**.

Selecting **Erase buffer before loading** tells the program to erase all buffer data using entered Erase value. Buffer erase is performed immediately before reading file content to buffer and it is functional for binary and all HEX file formats. Using this one-shot setting disables current setting of **Erase buffer before loading** option in menu **Options / General options** at tab **Hex file options**.

If the checkbox **Swap bytes** is displayed, the user can activate the function of swapping bytes within 16bit words (or 2-byte words) during the reading of a file. This feature is useful especially when loading files with Motorola representation of byte order in file (big endian). Standard file loading uses little endian byte order.

Note: *Big-endian and little-endian are terms that describe the order in which a sequence of bytes are stored in computer memory. Big-endian is an order in which the "big end" (most significant value in the sequence) is stored first (at the lowest storage address). Little-endian is an order in which the "little end" (least significant value in the sequence) is stored first. For example, in a big-endian computer, the two bytes required for*

the hexadecimal number 4F52 would be stored starting at storage address 1000H as: 4FH is stored at address 1000H, and 52H will be at address 1001H. In a little-endian system, it would be stored as 52H at address 1000H, and 4FH at address 1001H.

Number 4F52H is stored in memory:

Address	*Big endian system*	*Little endian system*
1000H	*4FH*	*52H*
1001H	*52H*	*4FH*

The reserved key **<F3>** will bring out this menu from any menu and any time.

File / Save

This command saves data in the buffer, which has been created, modified, or read from a device onto a specified file. The file format of saved file can be chosen from supported formats list box. The Buffer start and Buffer end addresses can also be specified, selecting the part of the buffer to save to the file. Supported file formats now are **binary**, **MOTOROLA**, **MOS Technology**, **Tektronix**, **Intel (extended) HEX**, **ASCII space**, **JEDEC** and **POF**.

If the checkbox **Swap bytes** is displayed, the user can activate the function of swapping bytes within 16bit words (or 2-byte words) during writing to file. This feature is useful especially when saving files with Motorola representation of byte order in file (big endian). Standard save file operation is using little endian byte order.

The reserved key **<F2>** will bring out this menu from any menu and any time.

File / Load project

This option is used for loading the project file, which contains device configuration buffer data saved and user interface configuration.

The standard dialog **Load project** contains additional window - **Project description** - placed at the bottom of dialog. This window is for displaying information about the currently selected project file.
Project information consists of:

- manufacturer and name of the first device selected in the project
- date and time of project creation

- user written description of project (it can be arbitrary text, usually author of project and some notes)

Note: *for projects with serialization turned on*

Serialization is read from project file by following procedure:

1. *Serialization settings from project are accepted*
2. *Additional serialization file search is performed. If the file is found it will be read and serialization settings from the additional file will be accepted. Additional serialization file is always associated to the specific project file. When additional serialization file settings are accepted, project serialization settings are ignored.*

Name of additional serialization file is derived from project file name by adding extension ".sn" to project file's name.
Additional serialization file is always placed to the directory "serialization\" into the control program's directory.

Example:

Project file name: my_work.prj
Control program's directory: c:\Program Files\Programmer

The additional serialization file will be:
c:\Program Files\Programmer\serialization\my_work.prj.sn

Additional serialization file is created and refreshed after successful device program operation. The only requirement for creating additional serialization file is load project with serialization turned on.

*Command **File / Save project** deletes additional serialization file, if the file exists, associated with currently saved project.*

File / Save project

This option is used for saving the project file, which contains settings of device configuration and buffer data saved. Data saved to project file can be restored anytime by menu command **File / Load project**.

The dialog **Save project** contains three additional windows in **Project description** panel placed at the bottom of dialog **Save project**. The windows are for displaying information about currently selected project file in dialog **Save project** and information about current project, which has to be saved. Dialog **Save project** contains also additional button with picture of key displayed. Clicking on this button password dialog appears which can be used to save project with password. Projects with password are special projects also called **Protected mode projects**. For more detailed

information about project passwords see **Options / Protected mode**.

Project information consists of:

- manufacturer and name of the first device selected in the project
- date and time of project creation
- user written description of project (it can be arbitrary text, usually author of project and some notes)

The first (upper) window contains information about currently selected project file in dialog Save project.

The second (middle) windows displays information about actual program configuration including currently selected device, active programmer, date, time. These actual program settings are used for creation of project description header.

The third (bottom) window is user editable and contains project description (arbitrary text), which usually consists of project author and some notes.

File / Reload file

Choose this option to reload a recently used file.

When you use a file, it is added to the **Reload file** list. Files are listed in order depending on time of use of them. Lastly used files are listed before files used far off.

To Reload a file:

1. From the File menu, choose Reload file.
2. List of lastly used files is displayed. Click the file you want to reload.

Note: *When reloading a file the file format is used, by which the file was lastly loaded/saved.*

File / Reload project

Choose this option to reload a recently used project.

When you use a project, it is added to the **Reload project** list. Projects are listed in order depending on time of use of them. Lastly used projects are listed before projects used far off.

To Reload a project:

1. From the File menu, choose Reload project.
2. List of lastly used projects is displayed. Click the project you want to reload.

File / Project options

This option is used to display/edit project options of the actually loaded project. Project options means basic description of project including following project data:

- device name and manufacturer
- project creation date
- user defined project description (arbitrary text), e.g. project author and other text data for more detailed project description

User can directly edit user defined project description only. Device name, manufacturer, project date and program version are generated automatically by program.

File / Load encryption table

This command loads the data from binary file from disk and it saves them into the part of memory, reserved for an encryption (security) table.

File / Save encryption table

This command writes the content of the memory's part, reserved for an encryption table, into the file on the disk as a binary data.

File / Exit without save

The command deallocates heap, cancels buffer on disk (if exists) and returns back to the operation system.

File / Exit and save

The command deallocates heap, cancels buffer on the disk (if exists), saves current setting of last 20 selected devices to disk and returns back to the operation system.

Buffer

Menu **Buffer** is used for buffer manipulation, block operation, filling a part of buffer with string, erasing, checksum and of course editing and viewing with other items (find and replace string, printing...).

Buffer / View/Edit

This command is used to view (view mode) or edit (edit mode) data in the buffer (for viewing in DUMP mode only). Use the arrow keys to select the data for edit. Data to be edited is signified by colour.

You can use <F4> as the hot key also.

View/Edit Buffer

F1	display help of actual window
F2	fill block causes filling selected block of buffer by requested hex (or ASCII) string. Sets start and end block for filling and requested hex or ASCII string.
Ctrl+F2	erase buffer with specified blank value
Ctrl+Shift+F2	fill buffer with random data
F3	copy block is used to copy specified block of data in current buffer on new address. Target address needn't be out from source block addresses.
F4	move block is used to move specified block of data in current buffer on new address. Target address needn't be out from source block addresses. Source address block (or part) will be filled by topical blank character.
F5	swap bytes command swaps a high- and low- order of byte pairs in current buffer block. This block must started on even address and must have an even number of bytes. If these conditions do not fulfil, the program modifies addresses itself (start address is moved on lower even address and/or end address is moved on higher odd address).
F6	print buffer
F7	find string (max. length 16 ASCII characters)
F8	find and replace string (max. 16 ASCII chars.)
F9	change current address
F10	change mode view / edit
F11	switch the mode of buffer data view between 8 bit and 16 bit view. It can be also do by mouse clicking on the button to the right of View/Edit mode buffer indicator. This button indicates actual data view mode (8 bit or 16 bit), too.
F12	checksum dialog allows to count checksum of selected block of buffer change mode view / edit
Arrow keys	move cursor up, down, right and left
Home/End	jump on start / end current line
PgUp/PgDn	jump on previous / next page
Ctrl+PgUp/PgDn	jump on start / end current page
Ctrl+Home/End	jump on start / end current device
Shift+Home/End	jump on start / end current buffer

Backspace move cursor one position left (back)

Note: *characters 20H - FFH (mode ASCII) and numbers 0..9, A..F (mode HEX) immediately changes content of edit area.*
Warning*: Editing of ASCII characters for word devices is disabled.*

Print buffer

This command allows writing the selected part of the buffer to a printer or file. The program uses an external text editor in which the selected block of the bufferdata is displayed and then printed or saved. By default, the text editor is set to **Notepad.exe**, which is standard part of all versions of Windows.

The Print Buffer dialog box options are:

Block start
Defines start address of selected block in buffer.
Block end
Defines end address of selected block in buffer.

External editor
This item defines the path and name of an external program, which can be used as a text viewer. By default it is set to Notepad.exe, which is standard in all versions of Windows. The user can define any text editor, for example Wordpad.exe, which is able to work with larger text files. The user defined text editor can be used to print or save the selected block of bufferdata. The external editor path and name is saved automatically to disk.

Find dialog box

Enter the search string in either text or Ascii and choose **<Find>** to begin the search.

Direction box specifies which way you want to search, starting from the current cursor position (In edit mode). **Forward** (from the current position or start of buffer to the end of the buffer) is the default. **Backward** searches toward the beginning.

Origin specifies where the search should start.

Find & Replace dialog box

Enter the search string in the **Text to find** input box and enter the replacement string in the **Replace with** input box.

The **Options** box allows you to select prompt on replace: if program finds the string, you will be asked before the program changes it.

Origin specifies where the search should start.

Direction box specifies which way you want to search, starting from the current cursor position (In edit mode). **Forward** (from the current position or start of buffer to the end of the buffer) is the default. **Backward** searches toward the beginning. In view mode searches all buffer.

Press **<Esc>** or click **Cancel** button to close dialog window.

By pressing **Replace** button the dialog box is closed and a Question window is displayed. This window contains following choices:

Yes	replaces found item and finds next
No	finds next item without replacing current one
Replace All	replaces all found items
Abort search	aborts this command

View/Edit buffer for PLD

Ctrl+F2	erase buffer with specified blank value
Ctrl+Shift+F2	fill buffer with random data
F9	go to address...
F10	change mode view / edit
F11	switch the mode of buffer data view between 1 bit and 8 bit view. It can be also do by mouse clicking on the button to the right of View/Edit mode buffer indicator. This button indicates actual data view mode (1 bit or 8 bit), too.
Arrow keys	move cursor up, down, right and left
Home/End	jump on start / end current line
PgUp/PgDn	jump on previous / next page
Ctrl+PgUp/PgDn	jump on start / end current page
Ctrl+Home/End	jump on start / end edit area
Backspace	move cursor one position left (back)

Note: *Characters 0 and 1 immediately changes content of edit area.*

Buffer / Fill block

This command fills the selected block of the buffer with the hex or Ascii string. Enter the block start and block end address as required. The default is the size of the selected device.

Buffer / Copy block

This command is used to copy a specified block of data to a new address within the current buffer space.

Buffer / Move block

This command is used to move a specified block of data to a new address within the current buffer space. Source address block (or part of) will be filled by the blank character.

Buffer / Swap block

This command swaps the high- and low- order of byte pairs in current buffer block. This block must start on even address and must have an even number of bytes. If these conditions are not fulfilled, the program modifies the addresses itself (start address is moved to a lower even address and/or end address is moved to a higher odd address).

Buffer / Erase

This command fills the content of the buffer with the blank character.

The key **<Ctrl+F2>** is the hot key.

Buffer / Fill random data

If this This command fills the content of the buffer with random data.

The reserved key **<Shift+Ctrl+F2>** is the hot key.

Buffer / Duplicate buffer

This command duplicates the buffer content in the source EPROM address range to the address range of the destination EPROM. For example, this command will copy the address range of a 27C256 three times to fill address range of a 27C010 EPROM.

Note: *The procedure always uses buffer start address 00000h.*

Buffer / Checksum

The checksum dialog is used for calculate checksums of selected block(s) in the buffer. The checksums are calculated as such :

Byte	sum by bytes to "word". CY flag is ignored
Word	sum by words to "word". CY flag is ignored
Byte (CY)	sum by bytes to "word". CY flag is added to result.
Word (CY)	sum by words to "word". CY flag is added to result.

CRC-CCITT	sum by bytes to "word" using RESULT=PREVIOUS + (x^16 + x^12 + x^5 +1)
CRC-XModem	sum by bytes to "word" using RESULT=PREVIOUS + (x^16 + x^15 + x^2 +1)

Column marked as **Neg.** is a negation of checksum so, that Sum + Neg. = FFFFH.
Column marked as **Suppl.** is complement of checksum so, that Sum + Suppl. = 0 (+ carry).

Dialog checksum contains following items:

From address: This is a start address of the block selected for calculating checksums in the buffer. Address is defined as Byte address.

To address: This is the end address of block selected for calculating checksums in the buffer. Address is defined as Byte address.

Insert checksum: When selected, the checksum will be written into the buffer. (see: **Calculate & insert)**

Insert at address: This specifies an address within the buffer where the chosen checksum will be written. (see: **Calculate & insert)** The Address can not be within the range specified by **<From address>** to **<To address>.** Address is defined as Byte address.

Size: This item is used for setting a size of chosen checksum result, which will be written into the buffer. The size of checksum result may be 8 (byte) or 16 (word) bits long. If word size was selected, whole checksum value will be written into the buffer.
Note: *If word size was selected, a low byte of checksum value will be written on address specified in box Insert address and a high byte will be written on address incremented by one.*

Calculate: Start calculating checksums for selected block in the buffer. No writes into the buffer are executed.

Calculate & insert: Start calculating checksums for selected block in the buffer and writes the chosen checksum into the buffer on address specified by **Insert address**.

Device

The Device Menu includes functions for working with the selected programmable device. Such as: device select, read data from device, device blank check, device program, device verify and device erase.

Device / Select from default devices

This window allows selecting one of the previously used devices. This window is a cyclic buffer in which are stored the last 20 selected devices including its device options. This list is saved to disk by command **File / Exit and save**.

If you wish display additional informations about the current device, use an **<Ctrl+F1>** key. This command will display the device size, organization, programming algorithm and a list of programmers (including auxiliary modules) that support this device. Package information and other general information about current device is also available..

Use the **<Del>** key for delete of current device from the list of default devices. You can delete all but the last device used from the list.

Device / Select device ...

This window allows selecting a device from all the devices supported. It is possible to sort by **name**, by **type** or by **manufacturer**.

The selected device is automatically saved to a buffer of default devices (max. 20 devices). This buffer is accessible with **Device / Select from default devices** command.

If you wish to display additional information about the current device, use an **<Ctrl+F1>** key. This command displays the size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device. Package and other general information about the current device are also available.

Select device ... / All

This window allows selecting the desired device from all devices supported by the programmer. Supported devices are displayed in a list box.

Device can be selected by double clicking the line on which the device is on or by entering manufacturer name and/or device number in a search box (use a key **<Space>** as a separation character) and press **<Enter>** or click **OK** button.

Press **<Esc>** or click **Cancel** at any time to cancel device selection without affecting the currently selected device.

The selected device is automatically saved to buffer of default devices (max. 20 devices). This buffer is accessible with **Device / Select from default devices** command.

If you wish display additional information about the current device, use an **<Ctrl+F1>** key. This command displays the size of the device, organization, programming algorithm and a list of programmers (including auxiliary modules), which supported this device. You can find here Package and other general information about the selected device is also available.

Select device ... / Only selected type

This window allows selecting the desired device type. First - you must select a device type (e.g. EPROM, Prom, PLD) and device subtype (e.g. 64Kx8 (27512)), using mouse or cursor keys. A list of manufacturers and devices will then be displayed.

Device can be selected by double clicking the line on which the device is on or by entering manufacturer name and/or device number in a search box (use a key **<Space>** as a separation character) and press **<Enter>** or click **OK** button.

Press **<Esc>** or **Cancel** at any time to cancel device selection without affecting the currently selected device.

Selected device is automatically saved to the buffer of default devices (max. 20 devices). This buffer is accessible with **Device / Select from default devices** command.

If you wish to display additional information about the current device, use the **<Ctrl+F1>** key. This command displays the size of the device, organization, programming algorithm and a list of programmers (including auxiliary modules) that support this device. Package and other general information about the selected device is also available.

Select device ... / Only selected manufacturer

This window displays the desired device by manufacturer. First select a required manufacturer in Manufacturer box using mouse or cursor keys.

Select the Device by double clicking the line the desired device is on or by entering device number in the search box (use a key **<Space>** as a separation character) and press **<Enter>** or click **OK** button.

Press **<Esc>** or click **Cancel** at any time to cancel device selection without affecting the currently selected device.

The selected device is automatically saved to buffer of default devices (max. 20 devices). This buffer is accessible with **Device / Select from default devices** command.

If you wish display additional information about the current device, use an **<Ctrl+F1>** key. This command displays the size of device, organization, programming algorithm and a list of

programmers (including auxiliary modules) that supports this device. Package and other general information about the device is also available.

Device / Select EPROM /Flash by ID

Use this command to autoselect a EPROM or Flash by reading the device ID. The programmer can automatically identify certain devices by reading the manufacturer and the device-ID that are burnt into the chip. This only applies to EPROM or Flash that supports this feature. If the device does not support a chip ID and manufacturer's ID, a message will be displayed indicating this as an unknown or not supported device.

If more devices with identical chip ID and manufacturer's ID were detected, the list of these devices will be displayed. A corresponding device can be chosen from this list by selecting its number (or manufacturer name) and press **<Enter>** (or click **OK** button). Press a key **<Esc>** or click **Cancel** button at any time to cancel device selection without affecting the currently selected device.

Warning: *The control program only support EPROM's and Flash with 28 and 32 pins at this time. Any of programmers determines pins number automatically. For other programmers you must enter this number manually.*

The programmer applies a high voltage to the appropriate pins on the socket. This is necessary to enable the system to read the device ID. Do not insert into the socket a device that is not an EPROM or Flash. It may be damaged when the programmer applies the high voltage.

We don't recommend apply this command to 2764 and 27128 EPROM types, because most of them ID not supports.

Device / Device options

All settings of this menu are used for programming process, serialization and associated file control.

Device / Device options / Operation options

All settings of this command are used for programming process control. This is a flexible environment, which content items associated with current device and programmer type. Items, which are valid for the current device but aren't supported by current programmer, are disabled. These settings are saving to disk along with associated device by **File / Exit and save** command.

The commonly used term are also explained in the user's manual to programmer. The special terms used here are exactly the terms used by manufacturer of respective chip.

Please read the documentation to the chip you want to program for explanation of all used terms.

List of commonly used items:

group **Addresses**:

device start address (default 0)
device end address (default device size-1)
buffer start address (default 0)
Split (default none)

This option allows to set special mode of buffer when programming or reading device. Using split options is particularly useful when using 8-bit data memory devices in 16-bit or 32-bit applications.
Following table describes buffer to device and device to buffer data transfer

Split type	Device	Buffer Address assignment
None	Device[ADDR]	Buffer[ADDR]
Even	Device[ADDR]	Buffer[2*ADDR]
Odd	Device[ADDR]	Buffer[1+(2*ADDR)]
1./4	Device[ADDR]	Buffer[4*ADDR]
2./4	Device[ADDR]	Buffer[1+(4*ADDR)]
3./4	Device[ADDR]	Buffer[2+(4*ADDR)]
4./4	Device[ADDR]	Buffer[3+(4*ADDR)]

Real addressing will be following: (all addresses are hexadecimal)

Split type	Device addresses	Buffer addresses
None	00 01 02 03 04 05	00 01 02 03 04 05
Even	00 01 02 03 04 05	00 02 04 06 08 0A
Odd	00 01 02 03 04 05	01 03 05 07 09 0B
1./4	00 01 02 03 04 05	00 04 08 0C 10 14
2./4	00 01 02 03 04 05	01 05 09 0D 11 15
3./4	00 01 02 03 04 05	02 06 0A 0E 12 16
4./4	00 01 02 03 04 05	03 07 0B 0F 13 17

Terms explanation:
Access to device address ADDR is written as Device[ADDR].
Access to buffer address ADDR is written as Buffer[ADDR].
ADDR value can be from zero to device size (in bytes).
All addresses are byte oriented addresses.

group **Insertion test:**

insertion test (default ENABLE)

If enabled, the programmer checks all pins of the programmed chip, if have proper connection to the ZIF

socket (continuity test). The programmer is able to identify the wrong contact, misinserted chip and also (partially) backinserted chip.

check ID bytes (default ENABLE)

If enabled, the programmer checks the electronic ID of the programmed chip.

Note 1: *Some old chips don't carry electronic ID*.

Note 2: *In some special cases, several microcontrollers don't provide ID, if copy protection feature in the chip is set, even if device ID check setting in control program is set to "Enable".*

group **Command execution:**

blank check before programming	(default DISABLE)
erase before programming	(default DISABLE)
verify after reading	(default ENABLE)
verify	(ONCE, TWICE)
verify options	(nominal VCC +/-5% nominal VCC +/-10% VCCmin - VCCmax)

group **ISP Target Supply Parameters**

Enable target system power supply - enables supplying of target system from programmer. Supply voltage for target system is switched on before action with programmed device and is switched off after action finished. If Keep ISP signals at defined level after operation is enabled, then programmer will switch off supply voltage after pull-up/pull-down resistors are deactivated.

Voltage - supply voltage for target system.

Max. current - maximum current consumption of powered target system.

Voltage rise time - determines skew rate of rising edge of target supply voltage (switch on supply voltage).

Target supply settle time - determines time, after which must be supply voltage in target system stabilized at set value and target system is ready to any action with programmed device.

Voltage fall time - determines skew rate of falling edge of target supply voltage (switch off supply voltage).

Power down time - determines time after switch off target system power supply within target system keeps residual supply voltage (e.g. from charged capacitor). After this time

elapsed target system has to be without supply voltage and can be safely disconnected from programmer.

group **Target System Parameters**

Oscillator frequency (in Hz) - oscillator's frequency of device (in target system). Control program sets programming speed by its, therefore is necessary set correct value.

Supply voltage (in mV) - supply voltage in target system. Control program checks or sets (it depends on programmer type) entered supply voltage in target system before every action on device.

Disable test supply voltage - disables measure and checking supply voltage of programmed device, set in Supply voltage edit box, before action with device.

Delay after reset active - this parameter determine delay after Reset signal active to start action with device. This delay depends on values of used devices in reset circuit of device and can be chosen from these values: 10ms, 50ms, 100ms, 500ms or 1s.

Inactive level of ISP signals - this parameter determine level of ISP signals after finishing access to target device. Signals of ISP connector can be set to Pull-up (signals are tied through 22k resistors to supply voltage) or Pull-down (signals are tied through 22k resistors to ground).

Keep ISP signals at defined level after operation - enables keeping set level of ISP signals after access to target device finished. Control program indicates activated pull-up/pull-down resistors by displaying window with warning. After user close this window control program will deactivate resistors.

Device / Device options / Serialization

Serialization is special mode of program. When a serialization mode is activated, a specified value is automatically inserted on predefined address into buffer before programming each device. When more devices are programmed one by one, the serial number value is changed for each device automatically and inserted into buffer before programming device, so each device has unique serial number.

There are two types of serialization:

- Incremental mode
- From file mode

If a new device is selected, the serialization function is set to a default state i.e. disabled.

Actual serialization settings for actually selected device are saving to disk along with associated device by **File / Exit and save** command.
When incremental mode is active following actual settings are saved to configuration file: address, size, serial value, incremental step and settings of modes ASCII / BIN, DEC / HEX, LS byte / MS Byte first.

When from-file mode is active following actual settings are saved to configuration file: name of input serialization file and actual label, which indicates the line with actual serial number in input file.

When program is in multiprogramming mode (multiple socket programmer is actually selected) the special section - **Action on not programmed serial values due to error** - is displayed in dialog **Serialization**. In this section two choices are available:

1. Ignore not programmed serial values
2. Add not programmed serial values to file

Ignore not programmed serial values means the not programmed serial values are ignored and no action is done with them.

Add not programmed serial values to file means the not programmed serial values are added to file. The file of not programmed serial values has the same text format as serialization file for ”From-file” serialization mode. So there is possible to program the serial values later on by ”From-file” serialization mode.

If device programming is stopped by user, program will not change the serial values ready for next batch of devices. The same situation is if device program is incomplete, e.g. for device insertion test error.

Ignoring or writing not programmed serial values is only used when at least one device from current batch of devices in multiple socket module programmer is completely programmed and verified without errors.

Note: *Serialization can work with control program’s main buffer only. It means the serialization can be used for device areas placed inside control program’s main buffer. Device special areas placed outside the program’s main buffer could not use serialization feature.*

Device / Device options / Serialization / Incremental mode

The **Incremental mode** enables to assign individual serial numbers to each programmed device. A starting number entered by user will be incremented by specified step for each device program operation and loaded in selected format to specified buffer address prior to programming of each device.

There are following options, that user can modify for incremental mode:

S / N size
S / N size option defines the number of bytes of serial value which will be written to buffer. For Bin (binary) serialization modes values 1-4 are valid for S / N size and for ASCII serialization modes values 1-8 are valid for S / N size.

Address
Address option specifies the buffer address, where serial value has to be written. Note that address range must be inside the device start and device end addresses. Address must be correctly specified so the last (highest or lowest) byte of serial value must be inside device start and device end address range.

Start value
Start value option specifies the initial value, from which serialization will start. Generally, the max. value for serialization is $1FFFFFFF in 32 bit long word.
When the actual serial value exceeds maximum value, three most significant bits of serial number are set to zero. After this action the number is always inside 0..$1FFFFFFF interval (this is basic style of overflow handling).

Step
Step options specify the increment step of serial value incrementation.

S / N mode
S / N mode option defines the form in which serial value has to be written to buffer. Two options are available:

- ASCII
- Bin

ASCII - means the serial number is written to buffer as ASCII string. For example number $0528CD is in ASCII mode written to buffer as 30h 35h 32h 38h 43h 44h (‘0’ ‘5’ ‘2’ ‘8’ ‘C’ ‘D’), i.e. six bytes.
Bin - means the serial number is written directly to buffer. If the serial number has more than one byte length, it can be written in one of two possible byte orders. The byte order can be changed in „Save to buffer“ item.

Style
Style option defines serial number base. There are two options:

- Decimal
- Hexadecimal.

Decimal numbers are entered and displayed using the characters '0' through '9'.
Hexadecimal numbers also use characters 'A' through 'F'.
The special case is Binary Dec, which means BCD number style. BCD means the decimal number is stored in hexadecimal number, i.e. each nibble must have value from 0 to 9. Values A to F are not allowed as nibbles of BCD numbers.
Select the base in „Style“ options before entering numbers of serial start value and step.

Save to buffer
Save to buffer option specifies the serial value byte order to write to buffer. This option is used for Bin S / N mode (for ASCII mode it has no effect).

Two options are available:

- LSByte first (used by Intel processors) will place the Least Significant Byte of serial number to the lowest address in buffer.
- MSByte first (used by Motorola processors) will place the Most Significant Byte first to the lowest address in buffer.

Split serial number at every N byte(s)
The option allows dividing serial number into individual bytes and placing the bytes at each Nth address of buffer. This feature is particularly useful for example for Microchip PIC devices when the device serial number can be the part of program memory as group of RETLW instructions. The example of using serial number split is listed in section Examples bellow as example number 2.

Examples:
1. Write serial numbers to AT29C040 devices at address 7FFFAH, size of serial number is 4 bytes, start value is 16000000H, incremental step is 1, the serial number form is binary and least significant byte is placed at the lower address of serial number in device.

   To make above described serialization following settings have to be set in Serialization dialog:

   Mode: Incremental mode
   S/N size: 4 bytes
   S/N mode:: Bin

Style: Hex
Save to buffer: LS Byte first
Address: 7FFFCH
Start value: 16000000H
Step: 1

Following values will be written to device:
The 1st device
Address Data
007FFF0 *xx xx xx xx xx xx xx xx xx xx xx* 00 00 00 16
The 2nd device
Address Data
007FFF0 *xx xx xx xx xx xx xx xx xx xx xx* 01 00 00 16
The 3rd device
Address Data
007FFF0 *xx xx xx xx xx xx xx xx xx xx xx* 02 00 00 16
etc.
"*xx*" mean user data programmed to device
Serial numbers are written to device from address 7FFFCH to address 7FFFFH because serial number size is 4 bytes.

2. Following example shows usage of SQTP serialization mode when serial number is split into RETLW instructions for Microchip PIC16F628 devices.
Note: *Serial quick turn programming (SQTP) is Microchip specified standard for serial programming of Microchip PIC microcontrollers. Microchip PIC devices allows you to program a unique serial number into each microcontroller. This number can be used as an entry code, password, or ID number.*
Serialization is done by using a series of RETLW (Return Literal W) instructions, with the serial number bytes as the literal data. To serialize, you can use Incremental mode serialization or From file mode serialization.

Incremental serialization offers serial number Split function. Serial number split allows usage of incremental numbers separated into even or odd bytes and between each byte of serial number RETLW instruction code is inserted.

From file serialization is using proprietary serial numbers file. This file can consist of various serial numbers. The numbers can have format suitable for SQTP that means number RETLW b1 RETLW b2 and so on. Note that PG4UW serial file format is not compatible with SQTP serial file generated by Microchip MPLAB.

Device PIC16F628 has 14 bit wide instruction word. Instruction RETLW has 14-Bit Opcode:

Description		MSB	14-Bit word	LSB
RETLW	Return with literal in W	11 01*xx*	*kkkk*	*kkkk*

where *xx* can be replaced by 00 and *k* are data bits, i.e. serial number byte

Opcode of RETLW instruction is hexadecimal 34*KK*H where *KK* is data Byte (serial number byte)

Let's assume we want to write serial number 1234ABCDH as part of four RETLW instructions to device PIC. The highest Byte of serial number is the most significant Byte. We want to write the serial number to device program memory at address 40H. Serial number split us very useful in this situation. Serialization without serial number split will write the following number to buffer and device:

AddressData
0000080 CD AB 34 12 *xx xx xx xx xx xx xx xx xx xx xx xx*

Note: *address 80H is because buffer has byte organization and PIC has word organization so it has equivalent program memory address 40H. When buffer has word organization x16, the address will be 40H and number 1234ABCDH will be placed to buffer as following:*

Address Data
0000040 ABCD 1234 *xxxx xxxx xxxx xxxx xxxx xxxx*

We want to use RETLW instruction so buffer has to be:

Address Data
0000040 34CD 34AB 3434 3412 *xxxx xxxx xxxx xxxx*

We can do this by following steps:

a) write four RETLW instructions at address 40H to main buffer (this can be done by hand editing buffer or by loading file with proper content). The bottom 8 bits of each RETLW instruction are not important now, because serialization will write correct serial number bytes at bottom 8 bits of each RETLW instruction.

   The buffer content before starting device program will look for example as following:

   AddressData
   0000040 3400 3400 3400 3400 *xxxx xxxx xxxx xxxx*

8 bits of each RETLW instructions are zeros, they can have any value.

b) Set the serialization options as following:
   S/N size 4 Bytes
   Address: 40H
   Start value: 1234ABCDH
   Step: 1
   S/N mode: BIN
   Style: HEX
   Save to buffer: LS Byte first

   Check the option "Split serial number at every N byte(s)" and split value N set to 2.
   (It means split of serial number to buffer at every second Byte)

   The correct serial number is set tightly before device programming operation starts.
   The buffer content of serial number when programming the first device is:

   Address Data
   0000040 34CD 34AB 3434 3412 *xxxx xxxx xxxx xxxx*

   That's it.

3. Following example uses the same serialization options as Example number 2, instead the serial number split is set to 3 and 4.

   When "Split serial number at every 3 byte(s)" is set, the buffer content will look as:

   Byte buffer organization:
   Address Data
   0000080 CD *xx xx* AB *xx xx* 34 *xx xx* 12 *xx xx xx xx xx xx*

   Word16 buffer organization:
   Address Data
   0000040 *xx*CD AB*xx xxxx xx*34 12*xx xxxx xxxx xxxx*

   When "Split serial number at every 4 byte(s)" is set, the buffer content will look as:

   Byte buffer organization:
   Address Data
   0000080 CD *xx xx xx* AB *xx xx xx* 34 *xx xx xx* 12

   Word16 buffer organization:
   Address Data
   0000040 *xx*CD *xxxx xx*AB *xxxx xx*34 *xxxx xx*12 *xxxx*

Advice: *When you are not sure about effects of serialization options, there is possible to test the real serial number, which will be written to buffer. The test can be made by following steps:*

1. *select wished serialization options in dialog Serialization and confirm these by OK button*
2. *in dialog Device operation options set Insertion test and Device ID check (if available) to Disabled*
3. *check there is no device inserted to programmer's ZIF socket*
4. *run Device Program operation (for some types of devices it is necessary to select programming options before programming will start)*
5. *after completing programming operation (mostly with some errors because device is not present) look at the main buffer (View/Edit buffer) at address where serial number should be placed*

Note: *Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device. If the buffer organization is byte org. (x8), the Serialization Address will be byte address. If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.*

Device / Device options / Serialization / From file mode

Using the From-file method, serial values are read from the user specified input file and written to buffer on address specified in input file.
There are two user options: **File name** and **Start label**.

File name
File name option specifies the file name from which serial addresses and values will be read. The input file for From file serialization must have special format, which is described in From file serialization file format below.

Start label
Start label defines the start label in input file. From defined start label starts reading of serial values from input file.

From file serialization file format
From file serialization input file includes addresses and arrays of bytes defining buffer addresses and data to write to buffer. Input file has text type format, which structure is:

[label*1*] addr byte*0* byte*1* .. byte*n*
...
[label*n*] addr byte*0* byte*1* .. byte*m* , addr byte*0* byte*1* ... byte*k*

_____/ _____/
| |
basic part optional part

; Comment

meaning is:

basic part
Basic part defines buffer address and array of bytes to write to buffer. Basic part must be always defined after label in line.

optional part
Optional part defines the second array of bytes and buffer address to write to buffer. One optional part can be defined after basic part of data.

label*1*, label*n* - labels
Labels are identifiers for each line of input file. They are used for addressing each line of file. The labels should be unique. Addressing lines of file means, the required start label entered by user defines line in input file from which serial values reading starts.

addr -
Addr defines buffer address to write data following the address.

byte*0*..byte*n*, byte*0*..byte*m*, byte*0*..byte*k* -
Bytes arrays byte*0*..byte*n*, byte*0*..byte*m* and byte*0*..byte*k* are defining data, which are assigned to write to buffer. Maximum count of bytes in one data field following the address is 64 bytes. Data bytes are written to buffer from address addr to addr+n.
The process of writing particular bytes to buffer is:
byte*0* to addr
byte*1* to addr + *1*
byte*2* to addr + *2*
....
byte*n* to addr + *n*

Optional part is delimited from the first data part by character " , " (comma) and its structure is the same as in the first data part, i.e. address and following array of data bytes.

Characters with special use:

[ ] - labels must be defined inside square brackets

',' – character which delimiters basic part and optional part of data

';' - the semicolon character means the beginning of a comment. All characters from „;„ to the end of line are ignored. Comment can be on individual line or in the end of definition line.

Note:

- *Label names can contain all characters except '[' and ']'. The label names are analysed as non case sensitive, i.e. character 'a' is same as 'A', 'b' is same as 'B' etc..*
- *All address and byte number values in input file are hexadecimal.*
- *Allowed address value size is from 1 to 4 bytes.*
- *Allowed size of data arrays in one line is in range from 1 to 64 bytes. When there are two data arrays in one line, the sum of their size in bytes can be maximally 80 bytes.*
- *Be careful to set correct addresses. Address must be defined inside device start and device end address range. In case of address out of range, warning window appears and serialization is set to disabled (None).*
- *Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device. If the buffer organization is byte org. (x8), the Serialization Address will be byte address. If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.*

Example:

[nav*1*] A7890 78 89 56 02 AB CD ; comment1
[nav*2*] A7890 02 02 04 06 08 0A
[nav*3*] A7890 08 09 0A 0B A0 C0 ; comment2
[nav*4*] A7890 68 87 50 02 0B 8D
[nav*5*] A7890 A8 88 59 02 AB 7D

;next line contains also second definition
[nav*6*] A7890 18 29 36 42 5B 6D , FFFF6 44 11 22 33 99 88 77 66 55 16

; this is last line - end of file

In the example file six serial values with labels „nav*1*“, „nav*2*“, ...“nav*6*“ are defined. Each value is written to buffer on address $A7890. All values have size 6 bytes. The line with „nav*6*“ label has also second value definition, which is written to buffer on address $FFFF6 and has size 10 bytes, i.e. the last byte of this value will be written to address $FFFFF.

Note: *Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device. If the buffer organization is byte org. (x8), the Serialization Address will be byte address. If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.*

Device / Device options / Statistics

Statistics gives information about the actual count of device operationsfor a selected device. If one device is corresponding to one device operation, e.g. For example, the number of device operations will be equal to number of programmed devices.

The next function of statistics is **Count down**. Count down allows checking the number of device operations, and then number of devices, on which device operations have to be done. After each successful device operation the value of the count down counter is decremented. The initial Count down value is user defined, (see the Statistics window). When count down value reaches zero, the specified number of device operations are is completed. A message to the user will then be displayed.

Statistics dialog box contains following options:

Check boxes **Program**, **Verify**, **Blank**, **Erase** and **Read** define operations, after which statistics values increment.

Check box **Count down** sets Count down activity (enable or disable). Edit box following the Count down check box defines initial number of count down counter, from which count down starts.

Statistics dialog box can be also opened by pressing right mouse button on the Statistics panel and clicking Statistics.

Actual statistics values are displaying in main window of control program in Statistics panel.

Statistics panel contains three statistics values – **Success**, **Failure**, **Total** and two **Count down** values, **Count down** and **Remains**.

Meaning of the values is:

Success	number of operations which where successfully completed
Failure	number of operations which where not successfully completed
Total	number of all operations

Count down	informs about Count down activity (Enabled or Disabled)
Remains	informs about the number of devices remaining.

A successful operation means any device operation completed without errors:

- program
- verify
- blank check
- erase
- read

If device operation terminates with error(s), it is not successful operation.

When new device type is selected, all statistics values are set to zero and **Count down** is **Disabled**.
Reset button in the **Statistics** panel reset statistics values.
Reload Count down button in **Statistics** panel reloads initial value to **Count down**.

Device / Device options / Associated file

This command is used for associating a file with the current device. This is a file, which can be automatic loaded to the buffer after a device is selected.

You can edit the associated file name in the file name box, full pathname is required. If enabled, the control program checks the present of this file.

You can save both settings i.e. associated file and enabling of automatic load of this file to disk by command **File / Exit and save**.

Device / Device options / Special options

The special terms used here are exactly the terms used by manufacturer of respective chip. Please read the documentation to the chip you want to program for explanation of all used terms.

Device / Blank check

This command performs a blank check on a device or part of a device, if possible. The control program reports the result of this action by a message to the INFO window.

The menu command **Device / Device options / Operation options** allows the setting of parameters related to blank check device operations.

Device / Read

This command allows reading the entire device or part of it into the buffer. The control program displays a message in the INFO window when finished.

The menu command **Device / Device options / Operation options** allows the setting of parameters related to the read device operation. Setting the option Verify data after reading in this menu command means a higher reliability for device reading.

Device / Verify

This command compares the programmed data of the device with the data in the buffer. The control program displays the result of this action by a message to the INFO window.

The menu command **Device / Device options / Operation options** allows the setting of parameters related to the verify device operation.

In the menu **Options / Display errors,** the errors can be displayed on the screen, written to the file: VERIFY.ERR or disabled.

Device / Program

This command allows the programming of the device or part of it by the data in the buffer. The control program displays the result of this action by a message to the INFO window.

The menu command **Device / Device options / Operation options** allows the setting of parameters related to the program device operation.

Device / Erase

This command allows erasing of the entire programmable device. The control program displays the result of this action by a message to the INFO window.

Device / Test

This command executes a test on the selected device selected from list of supported devices (e.g. static RAM) on programmers, which support this test.

Device / IC test

This command activates a test section for ICs separated by type to any libraries (on distribution CD). First select an

appropriate library, wished device and then a mode for test vectors run (LOOP, SINGLE STEP). Control sequence and test results are displayed to LOG WINDOW. In case of need is possible to define the test vectors directly by user. Detailed description syntax and methods of creation testing vectors is described in example_e.lib file, which is in programs installation folder. Note. Because the rising/falling edges of programmers are tuned for programming of chips, it may happen the test of some chips fails, although the chips aren't defective (counters for example).

Device / JAM/VME/...Player

Jam STAPL was created by Altera® engineers and is supported by a consortium of programmable logic device (PLD) manufacturers, programming equipment makers, and test equipment manufacturers.

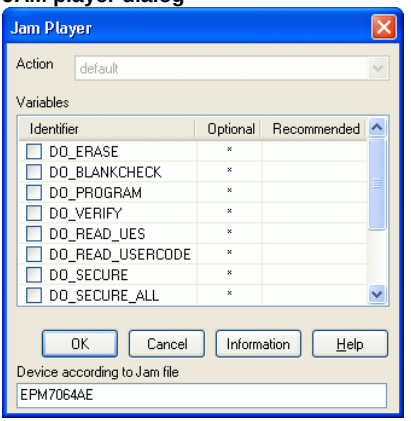
The Jam™ Standard Test and Programming Language (STAPL), JEDEC standard JESD-71, is a standard file format for ISP (In-System Programming) purposes. Jam STAPL is a freely licensable open standard. It supports programming or configuration of programmable devices and testing of electronic systems, using the IEEE 1149.1 Joint Test Action Group (JTAG) interface. Device can be programmed or verified, but Jam STAPL does not generally allow other functions such as reading a device.

The Jam STAPL programming solution consists of two components: Jam Composer and Jam Player.
The Jam Composer is a program, generally written by a programmable logic vendor, that generates a Jam file (.jam) containing the user data and programming algorithm required to program a design into a device.
The Jam Player is a program that reads the Jam file and applies vectors for programming and testing of devices in a JTAG chain.
The devices can be programmed in ZIF socket of the programmer or in target system through ISP connector. It is indicated by [PLCC44](Jam) or (ISP-Jam) suffix after name of selected device in control program. Multiple devices are possible to program and test via JTAG chain: JTAG chain (ISP-Jam)

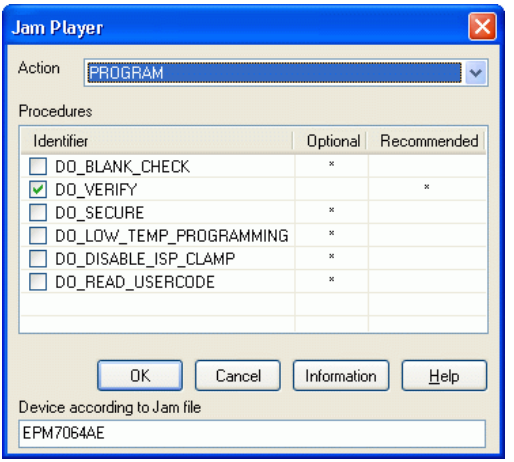
More information on the website:
http://www.altera.com/support/devices/programming/jam/dev-isp_jam.html
In-System Programmability Guidelines
http://www.altera.com/literature/an/an100.pdf
Using Jam STAPL for ISP & ICR via an Embedded Processor
http://www.altera.com/literature/an/an122.pdf

Software tools:
Altera: MAX+plus II, Quartus II, SVF2Jam utility (converts a serial vector file to a Jam file), LAT2Jam utility (converts an ispLSI3256A JEDEC file to a Jam file);
Xilinx: Xilinx ISE Webpack or Foundation software (generates STAPL file or SVF file for use by utility SVF2Jam);

JAM player dialog



Jam Player version 1 (see Action and Variables controls)



Jam Player version 2 (see Action and Procedures controls)

Action
Select the desired action for executing.

Jam file of version 2 consists of actions. Action consists of calling of procedures which are executed.
Jam file of version 1 does not know statements 'action' and 'procedure', therefore choice Action is not accessible. Program flow starts to run instructions according to boolean variables with prefix DO_something. If you need some new boolean variables with prefix DO_something then contact us.

Procedures
Program flow executes statements from each procedure. Procedures may be optional and recommended. Recommended procedures are marked implicitly. You can enable or disable procedures according to your needs. Jam Player executes only marked procedures. Other procedures are ignored. Number of procedures is different, it depends on Jam file.

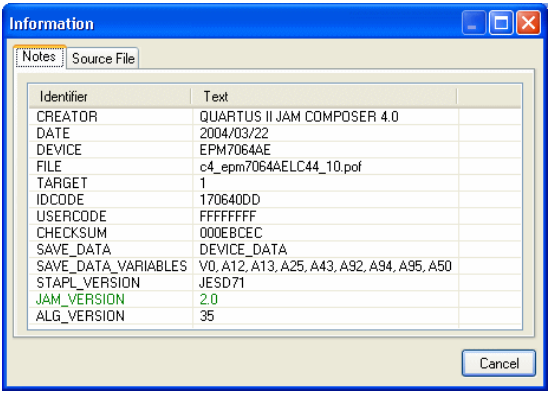
Variables
Jam file of version 1 does not know statements 'action' and 'procedure'. Program flow starts to run instructions according to boolean variables with prefix DO_something. Jam Player executes all marked DO_something cases in algorithm. Number of variables (procedures) is constant, it does not depend on Jam file. If you need some new boolean variables with prefix DO_something then contact us.

OK
Accept selected action with appropriate procedures which are marked.

Information
Displays informations about Jam file. You can preview NOTEs and source file in dialog.

Device according to Jam file
file is made for a specific device. Device name is found in Jam file in part NOTE identifier DEVICE. Device name must be identical with name of the device selected in dialog Select device. When devices are different, software will indicate this situation by warning message during start of the Jam Player.

JAM file information dialog



Notes
statements are used to store information about the Jam file. The information stored in NOTE fields may include any type of documentation or attributes related to the particular Jam program.

Source file
contains a program in Jam language. Jam program consists of a sequence of statements. Jam statement consists of a label, which is optional, an instruction, and arguments, and terminates with a semicolon (;). Arguments may be literal constants, variables, or expressions resulting in the desired data type (i.e., Boolean or integer). Each statement usually occupies one line of the Jam program, but this is not required. Line breaks are not significant to the Jam language syntax, except for terminating comments. An apostrophe character (') can be used to signify a comment, which is ignored by the interpreter. The language does not specify any limits for line length, statement length, or program size. More informations can be found on the website: http://www.altera.com/support/devices/programming/jam/dev-isp_jam.html.

Jam file with extension .jbc is Jam STAPL Byte code format which is not visible.

Converting JED file to Jam STAPL file for XILINX devices:

1. install Xilinx Integrated Software Environment (ISE) 6.3i software free download: WebPACK_63_fcfull_i.exe + 6_3_02i_pc.exe (315MB or so)
2. run Xilinx ISE 6/Accessories/iMPACT
   - in dialog “Operation Mod Selection: What do you want to do first?” choose: “Prepare Configuration Files”,

- in dialog “Prepare Configuration Files: I want create a:” choose: “Boundary-Scan File”,
- in dialog “Prepare Boundary-Scan File: I want create a:” choose: “STAPL File”,
- in dialog “Create a New STAPL File” write name of Jam file with extension .stapl,
- in dialog “Add Device” select JED file with extension .jed,
- in the created jtag chain select device e.g.: XC2C32A (left mouse button) and select sequence operation (e. g.: Erase, Blank, Program, Verify; right mouse button),
- in menu select item “Output/Stapl file/Stop writing to Stapl file”

3. run PG4UW, select device e.g.: Xilinx XC2x32A [QFG32](Jam), load Jam file (Files of type: select STAPL File)
4. choose “Device operation option Alt+O” press button “Jam configuration”. Warning “Select device from menu "Select Devices" and Jam file is probably different! Continue?” choose Yes. (Xilinx sw. does not include line: NOTE "DEVICE" "XC2x32A"; in Jam file). In dialog “Jam player” select action and procedures, finish dialogs, press button “Play Jam” from toolbar and read Log window

The ispVM Virtual Machine is a Virtual Machine that has been optimized specifically for programming devices which are compatible with the IEEE 1149.1 Standard for Boundary Scan Test. The ispVM EMBEDDED tool combines the power of Lattice's ispVM Virtual MachineTM with the industry-standard Serial Vector Format (SVF) language for Boundary Scan programming and test.

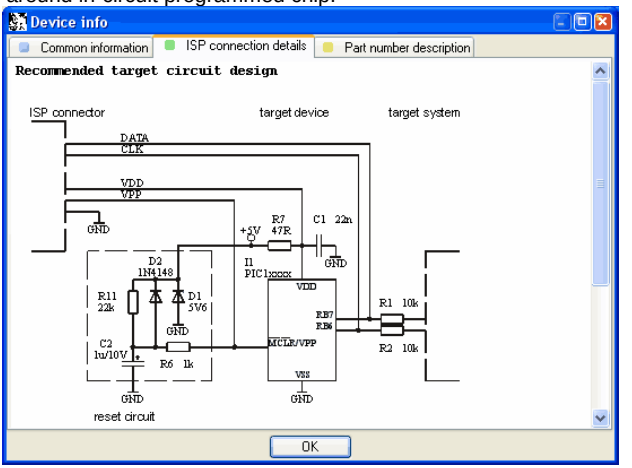
The ispVM System software generates VME files supporting both ispJTAG and non-Lattice JTAG files which are compliant to the IEEE 1149.1 standard and support SVF or IEEE 1532 formats. The VME file is a hex coded file that takes the chain information from the ispVM System window. The devices can be programmed in ZIF socket of the programmer or in target system through ISP connector. It is indicated by [PLCC44](VME) or (ISP-VME) suffix after name of selected device in control program. Multiple devices are possible to program and test via JTAG chain: JTAG chain (ISP-VME).

More information on the website:
http://www.latticesemi.com/products/devtools/software/ispvmembed/index.cfm
In-System Programmability Guidelines
http://www.latticesemi.com/products/technology/isp_usage.cfm

Software tools:
Lattice: ispLEVER, ispVM System ISP Programming Software, PAC-Designer Software, svf2vme utility (converts a serial vector file to a VME file)

Device / Device info

The command provides additional information about the current device - size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device. You can find the package information, part number description and full information for ISP implementation. For example: description of ISP connector pins for currently selected chip, recommended target design around in-circuit programmed chip.



The reserved key **<Ctrl+F1>** is the HOT key.

Programmer

Menu Programmer includes commands used for work with programmers.

Programmer / Find programmer

This menu is for programmer related operations. This command contains following items:

Programmer – Searches for an installed programmer. The control program will find all supported programmers.
Establish communication - allows manual or automatic establishing communication for a new programmer.

Speed - sets communication speed. If manual is selected, note that the speed is expressed as a percent from a maximal speed.

The communication speed modification is important for PCs with "slow" LPT ports, which haven't sufficient driving power for a PC<->programmer cable (laptop, notebook, ...). Use this command, if you have any communication problems with connected programmer on the LPT port of your PC (e.g. control program reports a programmer absence, the communication with the programmer is unreliable, etc.).

If automatic establishing communication is selected, then control program sets a maximal communication speed.

Port - selects a LPT port, which will be scanned for a requested programmer. If All port is selected, the control program scans all LPT ports, which are available on standard addresses.

Address for special port - sets address of LPT port, if a Special port is selected.

Pressing key **<Enter>** or button **OK** initiates scanning for programmer by set parameters. There is same activity as at start the control program. The command clears a list of default devices without the current device, if the new selected programmer supports this one.

This setting is saved to disk by command **Options / Save options**.

Programmer / Refind programmer

This menu command is used to reestablish communication withthe currently selected programmer.

To select other type of programmer, programmer communication parameters and to establish communication with newly selected programmer use the menu option **Programmer / Find programmer**.

Programmer / Handler

In dialog box, **Handler,** the Handler type and Handler communication parameters can be set. A Handler is an external device requiring special control of device related operations. When no Handler is selected, the default state, device operations are controlled directly by user. Otherwise, the handler mode of operation is enabled and device operations are controlled automatically in co-operation with the Handler.

Dialog **Handler** contains following items:

Selected Handler select wished Handler type.
Search at port select a COM port, which will be scanned for a requested Handler.

Pressing key **<Enter>** or button **OK** initiates scanning for Handler by set parameters. If selected Handler type is **None**, no Handler scanning will be processed. Current Handler settings are saved to configuration file by command **Options / Save options** or when control program is closed.
Handler is not available for sale.

Programmer / Module options

This option is used for multiple socket programmers for defining MASTER socket and activity of each socket. **MASTER socket** group box allows user to set socket which is preferentially used for device reading operation. **Enable/Disable socket** checkbox array allows user to set enabling and disabling of each socket individually. Disabled sockets are ignored for any device operation.

Programmer / Automatic YES!

This command is used for setting **Automatic YES!** mode. In this mode you just put a device into the ZIF socket and a last operation will be repeated automatically. Program automatically detects an insertion of a new device and runs last executed operation without pressing any key or button. An insertion of device into ZIF is displayed on the screen. Repeat last operation is cancelled by pressing the key **<ESC>** during waiting for insert/remove a device to/from ZIF.

After a device operation is executed, one of the OK or ERROR (status) LEDs on the programmer will light depending on the result of an operation and the BUSY LED will blinking.
If the program detects removal of a device, then status LED will switched off, but the BUSY LED will still blinking to indicate readiness of the program to repeat last operation with new device.
After the program determines that a new device is in the ZIF socket, the BUSY LED will goes to light continually. The programmer will wait for the device to be properly inserted, (ZIP socket closed). If this operations times out, the program will light the ERROR LED. After a new device is inserted correctly, the program will switch off all status LEDs, except BUSY, and will start an operation with new device.

This mode may be enabled or disabled by item **Automatic YES!** mode. If a new programmer is selected **Options / Find programmer**, this mode will be disabled.

Response time: allows the user to set the time interval for new device detection. Default is set standard interval. If a socket adapter is used, then it is recommended to set an elongated interval.

In **Pins with capacitors** bar may be entered a list of a pins interconnected by capacitors (for example: if a converter, which have connected capacitor between VCC and GND, is used), which may makes problems at detecting insertion of a new device.
List of pins of device is in form:
pinA, pinB, pinC....
Example: 4,6,17

In **Device removal hold off time** is the delay when the programmed device is removed and a new device is inserted into the programmer's ZIF socket. This interval is in seconds and must be from 1 to 120 (default value is 2 seconds).

In **Device insertion complete time** is the maximum time for all pins of the device to be properly inserted after the first pin(s) were detected so that the program does not report an insertion error. This interval is in seconds and must be from 1 to 120 (default value is 5 seconds).

This list is erased if a new device is selected by **Device / Select default** or **Device / Select device** ...

This setting is saved to disk by command **Options / Save options**.

Programmer / Selftest

This command executes a selftest of programmer without the diagnostic POD.

Programmer / Selftest plus

This command executes a selftest of the programmer using diagnostic POD, which is included with the programmer. We recommend running this test every 6 months.

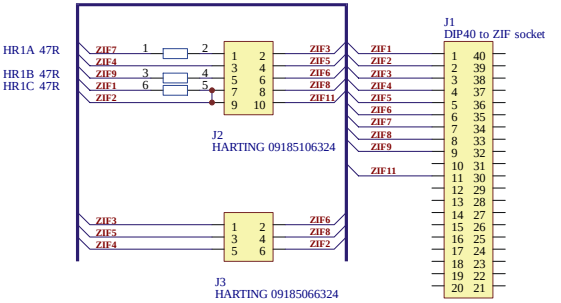
Programmer / Self test ISP connector

This command executes a selftest of the ISP connector for the programmer using the diagnostic POD for ISP connectors.

Diagnostic POD for ISP connectors is necessary to use for testing 6 and 10-pin ISP connectors of programmers.

Diagnostic POD for ISP is available as optional accessory for ISP-capable programmers. The order number: 70-0208

Schematic of Diagnostic POD for ISP connector (if you are in hurry):



Sequence for testing 6 pins ISP connector:

1. Insert Diagnostic POD for ISP connectors into ZIF socket of the programmer. Diagnostic POD must be inserted as 40 pins device.
2. Interconnect 6 pins connector of Diagnostic POD with an ISP connector of the programmer with an ISP cable, included in programmer delivery package. Be sure that pins are interconnected properly (i.e. 1-1, 2-2, ..., 6-6).
3. Run selftest of ISP connector in PG4UW (**Programmer / Selftest ISP connector**).

Sequence for testing 10 pins ISP connector:

1. Insert Diagnostic POD for ISP connectors into ZIF socket of the programmer. Diagnostic POD must be inserted as 40 pins device.
2. Interconnect 10 pins connector of Diagnostic POD with an ISP connector of the programmer with an ISP cable, included in delivery programmer package. Be sure that pins are interconnected properly (i.e. 1-1, 2-2, ..., 10-10).
3. Run selftest of ISP connector in PG4UW (**Programmer / Selftest ISP connector**).

We recommend run this test every 6 months.

Programmer / Calibration test

Command executes test of programmer's calibration values.

Programmer / Create diagnostic report

Create Diagnostic report is used for writing more specific diagnostic information to the **Log window** and consequently

copy **Log window** content to clipboard. The Log window content can be placed from clipboard to any text editor. Diagnostic reports are useful when error occurs and have to be documented. Diagnostic reports are useful when communicating with programmer or device manufacturers.

Options

The Options menu contains commands that let you view and change various default settings.

Options / General options

General options dialog allows users to control following program options:

File options

File options page allows you to set file masks, auto-reload of current file and choose file format.

File format mask is used for setting file-name masks to act as a filter for file listings in the **File / Save** and **File / Load** window. Masks may contain one of following wildcards (*, ?) .

Project file default extension is used for setting project file extension used as default in **File / Load project** and **File / Save project** dialog boxes.

If the current file is modified by another process, three options exist for reloading the actual file:

1. Prompt before reloading file
2. Reload automatically
3. Ignore change scanning of current file

There are three situations when file modification is tested:

- switching to the control program from another application
- selecting the device operation Verify or Program
- when repeat of last device operation is selected in dialog "Repeat?"

Load file format allows setting the file format recognition mode. When automatic file format is selected, program determines the format automatically. If file format matches one of supported formats, the file is read to buffer in the detected format.
Manual file format allows the user to select explicitly the file format. The file may be loaded incorrectly, if the file format does not match the user selected format.

Hex file options

This page contains several options for loading control by any of HEX formats.

The **first** option enables **erasing** of the buffer (with desired value) automatically before Reloading by any of the HEX formats.
The **second** option sets a **negative offset**, which is used for data address modifications by loading from any HEX file so, that data can be written to existing buffer addresses. Manual or Automatic negative offset mode can be set. We recommend automatic set of negative offset in special cases only. This option contain a heuristic analyze, which can treat some data in file incorrectly. There are especially critical files, which contain a fragmented addresses range and which exceeds a size of selected device - some block can be ignored. Automatic set of negative offset can be disabled by select of any special devices. No address range in files associated with special devices can be moved and no block can be removed from the file when reading the file. For special devices following negative offset options are available: Yes (negative offset is turned on) and No (negative offset is not used).

Example:
A file contents data in Motorola S - format. A data block started at address FFFF0H. It is a S2 format with length of address array of 3 bytes. For all data reading you can set a value of negative offset to FFFF0H. It means, that the offset will be subtracted from current real addresses and so data will be written from buffer address 0.

Warning: *The value of negative offset is subtracted from real address and therefore could result in a negative number. Therefore, be careful when selecting this value.*

Language

This page allows you to select another language for the user interface such as menus, buttons, dialogs, information and messages. It also allows the selection of help file in another language. For language support, a language definition file is required.

Sound

Sound page allows the user to select the sound mode of the program. Program generates sounds after some activities, such as: programming, verifying, reading, etc. Program generates sound when warning or error message is displayed. User can now select sound from Windows system sound (required installed sound card), PC speaker or none sound.

In the panel **Programmer internal speaker sound settings** is possible to set sound options for some programmers with built-in internal speaker. Sound beeps are then generated from internal programmer speaker after each device operation for indicating device operation result – good or bad result.

Log file

This options associates with using of **Log window**. All reports for Log window can be written into the Log file too. The Log file name is "Report.rep" as default. The control program creates this file with name and directory specified in Log file name edit box.
Following Log file options are available:

- **No** default, content of Log window is not copied to Log file, i.e. all reports will be displayed to Log window only
- **New** deletes old Log file and creates new one during each start of control program
- **Append** adds Log window reports into existing Log file, If file does not exist, the new file will be created

The Log file settings can be saved to disk by command **Options / Save options**.

Display errors

This option allows to set a form of error displaying as a result of programmed data verifying. Errors can be displayed to the screen (max. 45 differences), saved to error file of differences on the disk or it will not be displayed. In case the displaying errors are turned off, the control program reports a warning message in INFO window only. The default error file name is "Verify.err". The file name and directory can be user specified in edit box Error file name.

Following Display errors settings are available:

- **None** does not display error values on screen nor to the file
- **Screen** default, displays errors to Log window
- **File** writes error reports to error file

The Display errors settings can be saved to disk by command **Options / Save options**.

Save options

Page allows you to select the program options saving when exiting program. Three options are available here:

- **Don't save** don't save options during quitting program and don't ask for saving options
- **Auto save** save options during quitting program without asking for saving options
- **Prompt for save** program asks user for saving options before quitting program. User can select to save or not to save options

Other

Page **Other** allows user to manage other program settings.

Panel **Application priority** allows user to set the priority of the program. Priority settings can affect performance of programmer (device programming time), especially if there are running more demanding applications in the system. Please note that setting application priority level to Low can significantly slow down the program.

In the panel **Tool buttons**, hint display options on toolbar buttons in main program window can be modified. In the panel **Start-up directory** can be selected mode of selecting directory when program starts. **Default start-up directory** means directory, from which program is called. **Directory in which program was lastly ended** means the last current directory when program was lastly ended. This directory assumes the first directory from directory history list.

Options / View

Use the View menu commands to display or hide different elements of program environment such as toolbars.
Following toolbars are available now:

Options / View / Main toolbar

Choose this command to show or hide the Main toolbar.

Options / View / Additional toolbar

Choose this command to show or hide the Additional toolbar.

Options / View / Device options before device operation

Choose this command to enable/disable display of Device options before device operation is confirmed.

Options / Protected mode

Protected mode is special mode of program. When program is in Protected mode, there are disabled program operation and commands that can modify buffer or device settings. Protected mode is used for prevent operator from modify buffer or device settings due to insignificance. Protected mode is suitable for the programming of a large amount of the same type of devices.

There are two ways how to switch program to Protected mode:

1. by using menu command **Options / Protected mode**. This command displays password dialog. User has to enter password twice to confirm the password is correct. After password confirmation program switches to Protected mode. The entered password is then used to switch off Protected mode.
2. by reading project, that was previously saved in Protected mode. For details see **File / Save project**.

To switch program from Protected mode to normal mode, use the menu command **Options / Normal mode**. The "Password required" dialog appears. User has to enter the same password as the password entered during switch to Protected mode.

Other way to cancel Protected mode of program is closing of program, because program Protected mode is active until program is closed. The next program start will be to Normal (standard) mode (the only exception is case of project loaded by command line parameter name of project and the project was saved in Protected mode).

Options / Save options

This command saves all settings that are currently supported for saving, even if auto-save is turned off. Following options are saved: options under the Options menu, ten last selected devices, file history, main program window position and size.

Help

Pressing the **<F1>** key accesses the Help. When you are selecting menu item and press **<F1>**, you access context-sensitive help. If PG4UW is executing an operation with the programmer **<F1>** generates no response.

The following Help items are highlighted:
- words describing the keys referred to by the current Help
- all other significant words
- current cross-references; click on this cross-reference to obtain further information.

Since the Help system is continuously updated together with the control program, it may contain information not included in this manual.

Detailed information on individual menu commands can be found in the integrated on-line Help.

Note: *Information provided in this manual is intended to be accurate at the moment of release, but we continuously improve all our products. Please consult manual on* ***www.dataman.com***.

Help / Supported devices

This command displays list of all devices supported by at least one type of all supported programmers. It is useful especially when user wants to find any device supported by at least one type of programmers.
Prefix "g_" before name of device means the device is supported by multi-socket programmer.

Help / Supported programmers

This command displays information about programmers, where supported this program.

Help / Device list (current programmer)

This command makes a list of all devices supported by current programmer and saves it to **?????DEV.txt** text file and **?????DEV.htm** HTML file in the directory where control program is run from. Marks ????? are replaced by abbreviated name of current programmer, the device list is generated for.

Help / Device list (all programmers)

This command makes device lists for all supported programmers and saves them to **?????DEV.TXT** text files and **?????DEV.HTM** HTML files in the directory where control program is running from. Characters ????? are replaced by abbreviated name of programmers, the device lists are generated for.

Note: *The control program loses all information about current device after this command is executed. Reselect wished device again by any of select methods in menu* ***DEVICE****.*

Help / Device list (cross reference)

This command makes cross reference list of all devices supported by all programmers available on market and supported by this control program. The resulting list is in HTML format and consists of following files:

- one main HTML file **TOP_DEV.htm** with supported device manufacturers listed
- partial HTML files with list of supported devices for each device manufacturer

Main HTML file is placed to directory where this control program for programmers is located.

Partial HTML files are placed to subdirectory **DEV_HTML** placed to the directory where control program for programmers is located.

About

When you choose the Info command from the menu, a window appears, showing copyright and version information.

Common notes

Software

PG4UW is common control program for some Dataman programmers. Thus, during work with him it is possible to find some items, those refer not to current selected programmer.

Some special devices (e.g. Philips Coolrunner family) require external DAT files, that aren't present in standard PG4UW SW delivery on CD. If you need to program these devices, look at **www.dataman.com**, section Download.

You can start control program with different **command line parameters**.
Basic rules for using of executive command line parameters:

1. command line parameters are not case sensitive
2. command line parameters can be used when first starting of program or when program is already running
3. if program is already running, then any of command line operation is processed only when program was not busy (no operation was currently executing in program). Program must be in basic state, i.e. main program window focused, no modal dialogs displayed, no menu commands opened or executed.
4. order of processing command line parameters when using more parameters together is defined firmly as following:

    1. Load file (/Load file:...)
    2. Load project (/Prj:...)
    3. EPROM/Flash select by ID
    4. Program device (/Program[:switch])
    5. Close of control program (/Close only together with parameter /Program)

Available command line parameters:

/Axxx check programmer present on LPT port with address *xxx* only
example: /A3bc

/SPP force PC <-> programmer communication in unidirectional mode

Available executive command line parameters:

/Prj:<file_name> forces project load when program is starting or even if program is already running, <file_name> means full or relative project file path and name

/Loadfile:<file_name> forces file load when program is starting or even if program is already running, <file_name> means full or

relative path to file that has to be loaded, file format is detected automatically

/Program[:switch] forces start of ”Program device” operation automatically when program is starting, or even if program is already running, also one of following optional switches can be used:

switch 'noquest' forces start of device programming without question

switch 'noanyquest' forces start of device programming without question and after operation on device is completed, program doesn't show "Repeat" operation dialog and goes directly into main program window

Examples:

1. /Program
2. /Program:noquest
3. /Program:noanyquest

/Close this parameter has sense together with /Program parameter only, and makes program to close automatically after device programming is finished (no matter if operation was successful or no)

/Eprom_Flash_Autoselect[:xx] forces automatic select EPROM or FLASH by ID when program is starting or even if program is already running. xx means pins number of device in ZIF (this time are valid 28 or 32 pins only) and it is required just for older programmers without insertion test capability. For others programmers is the value ignored.

Hardware

Due a large variety of parallel port types, a case may occur when the programmer cannot "get concerted" with the PC. This problem may be shown as none communication between the PC and the programmer, or by unreliable communication. If this behaviour occur, try to connect your programmer to some other PCs or other parallel ports near you.

If you find none solution, please document the situation, i.e., provide us an accurate description of your PC configuration, including some other circumstances bearing on the problem in question, and advise the manufacturer of your problem. Don’t

forget please enter of PC type, manufacturer, speed, operation system, resident programs; your parallel port I/O manufacturer and type. Use please **Device problem report** form for this purpose (see **Appendix A**).

ISP (In-System Programming)

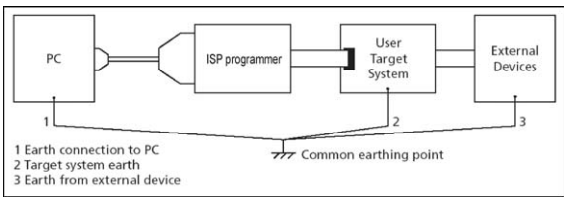
Definition

In-system programming allows programming and reprogramming of device positioned inside the end system. Using a simple interface, the ISP programmer communicates serially with the device, reprogramming nonvolatile memories on the chip. In-system programming eliminates the physical removal of chips from the system. This will save time and money, both during development in the lab, and when updating the software or parameters in the field.
Target device is the device (microcontroller, PLD, etc…), which is to be in-system programmed.
Target system is the physical Printed Circuit Board (PCB), which contains the device to be in-system programmed.
ISP programmer is programmer, which has in-system programming capability (for example Dataman-48Pro, Dataman-40Pro…).

General rules for in-system programming

We recommended respect following rules to avoid damage PC, ISP programmer, and target device or target system:

- Ensure common earth point for target system, ISP programmer and PC.
- For laptop or other PC that is not connected to common earth point: make hard - wired connection from laptop to common earth point (for example use LPT or COM port D – connector).
- Any devices connected to target system must be connected to common earth point too.



Direction of connect Dataman ISP programmer to target system:

During in-system programming you connect two electrical devices – ISP programmer and target system. Unqualified connection can damage these devices.

Note: *When you don't keep below directions and you damage programmer during in-system programming, it is damage of programmer by unqualified manipulation and is out of warranty.*

1. Turn off both devices – ISP programmer and target device.
2. Assign same GND potential for all devices, e.g. connect GND of all devices by wire.
3. Insert one connector of ISP cable to ISP programmer, turn on programmer and control program.
4. In control program select target device and operation options.
5. Start action on target device (read, program).
6. After direction of control program, connect other ISP cable connector to target system and turn on it.
7. After direction of control program, disconnect other ISP cable connector from target system and turn off it.
8. If you need another action on target device, you continue with step 5.

The recommendation for design of target system with ISP programmed device

The target system must be designed to allow all signals, which are use for In-system programming to be directly connected to ISP programmer via ISP connector. If target system use these signals for other function, is necessary isolated these signals. Target system mustn't affect these signals during In-system programming.

For in-system programmable devices manufacturers publish application notes. Design of Dataman programmers together with respect of these application notes allow proper In-system programming. Condition is exactly respect these application notes. Applications notes, which Dataman use in ISP programmers, are published in **www.dataman.com**, section Application notes.

Please, read some notes for following recommended circuits.

- *Purpose of D1 diode is to protect the target circuit against a higher voltage, which is provided by ISP programmer.*
- *If your target board supply differs from mentioned 5V, choose please the Zener diode (D1) voltage according to this supply voltage.*
- *We recommend to use resistors R1, R2, (R3) to separate the target device from target system. If pins needed for ISP programming are inputs in target system then separation by resistors is sufficient and resistors make a low pass filter*

too. If pins are outputs, then use of resistors saves a programming time. Of course the isolation resistors R1, R2, (R3) can be replaced by switches or jumpers, if necessary. In that case, during the ISP programming of target device the switches (jumpers) must be open. But the using of switches (jumpers) adds a next manipulation time to programming procedure.

Other

Attention to multitasking OS's (Windows 95/98/Me/NT/2000/XP). There is needful for regular running of control program for any Dataman programmer that printer port, on which is programmer connected, must be reserved for this programmer only. Otherwise, any other program must not simultaneously to use (or any way to modify) this printer port.

PG4UW SW can handle all modes of LPT port (full IEEE 1284 support), thus you don't need to configure LPT port for connection of Dataman programmers.

Please don't move **Info** window during BUSY LED is on - watching circuit can be activate to switch the programmer in safe status as in case communication PC-programmer error.

LPT port driver

For programmers connected through parallel LPT port, control program requires correctly installed LPT port driver. LPT port driver installation and uninstallation is made automatically by installation program. Normally there are no problems with the driver. But sometimes driver can not be initialized correctly. It is especially in the case that no LPT1 port is present in the Windows NT/2000/XP operating systems. When the LPT port driver is not initialized, control program can not detect any LPT ports in the system.

LPT driver requires port LPT1 to be present in the operating system. Please check the parallel port LPT1 is present in the system.

The short description, how to see LPT ports present in operating system:

1. click to "Start" menu
2. click with right mouse button to "My computer" item and select menu "Properties
3. in the "System properties" dialog select "Hardware" page and click to "Device manager" button

4. in the "Device manager" dialog select "Ports (Com & LPT)" (double click), it will show the list of all present LPT and COM ports

There should be displayed at least one present LPT port.

If there are present one or more LPT ports but with numbers other than LPT1, it is necessary to change one of the LPT ports to LPT1 port. Follow the steps bellow (continued from steps 1. - 4.)

5. double click to selected LPT port to show properties of the port
6. in the "LPT port properties" dialog select the page "Port settings"
7. change number of LPT port to LPT1 by "LPT Port Number" setting
8. click OK button
9. restart the operating system

(even if system does not require restart, it is necessary to perform system restart to correctly initialize our LPT port driver)

That's all. Our software should work properly with LPT connected programmer.

When using programmer connected through USB, there is no need of LPT port driver.

USB driver

For programmers connected through USB port, control program requires correctly installed USB driver.

We recommend to install control program first and then connect programmer to USB port. Windows will detect new hardware as USB programmer automatically.

When the programmer is connected to USB port before control program was installed, Windows will detect new hardware and ask user to select driver installation method: automatically or manually. To detect programmer correctly, control program installation CD must be inserted to computer's CD-ROM drive and following steps have to be done:

(driver installation steps bellow are used for Windows XP but other Windows versions have similar steps)

STEP 1
The first time a new USB device is plugged into a Windows XP system, a dialog box will appear indicating that the system has found a new hardware device. There may also be a dialog box

that informs the user that a device data base is being built or updated.

After these dialogs appear, the Found New Hardware Wizard dialog box is displayed. Select "Install from a list or specific location (Advanced)" and click "Next" to continue the installation.

STEP 2
Make sure that "Search for the best driver..." is selected. Select "Search removable media" and deselect "Include this location in the search". Click "Next".
STEP 2A
During the install, a dialog will pop up stating, "The software you are installing for this hardware...has not passed Windows Logo testing..." Click "Continue Anyway."

STEP 3
The "Completing the Found New Hardware Wizard" will appear once the programmer has been installed. Click "Finish" to end the USB installation.

Troubleshooting and warranty

Troubleshooting

We really want you to enjoy our product. Nevertheless, problems can occur. In such cases please follow the instructions below.

- It might be your mistake in properly operating the programmer or its control program PG4UW.
  - Please read carefully all the enclosed documentation again. Probably you will find the needed answer right away.
  - Try to install programmer and PG4UW on another computer. If your system works normally on the other computer you might have a problem with the first one PC. Compare differences between both computers.
  - Ask your in-house guru (every office has one!).
  - Ask the person who already installed the programmer.

- If the problem persists, please call the local dealer, from whom you purchased the programmer, or call Dataman direct. Most problems can be solved by phone, e-mail or fax. If you want to contact us by:
  - **Mail/fax** - Copy the "**DEVICE PROBLEM REPORT**" form and fill it in following the instructions at the end of the form. Write everything down that you consider being relevant about the programmer, software and the target device. Send the completed form by mail or fax to Dataman (fax number in the control program, menu **Help / About**) or to your local dealer. If you send the form by fax please use black ink, a good pen and large letters!
  - **E-mail** - Use "**DEVICE PROBLEM REPORT**" form from our internet site and fill it in following the instructions at the end of the form. Use standard ASCII editor. Write everything down that you consider being relevant about the programmer, software and the target device. Send the completed form by e-mail to your local dealer or to Dataman **support@dataman.com**).
  - **Phone** - Copy "**DEVICE PROBLEM REPORT**" form and fill it in following the instructions at the end of the form. Write everything down that you consider being relevant about the programmer, software and the target device. Send the completed form by mail or fax to Dataman (fax number in the control program, menu **Help / About**) or to your local dealer. If you send the form by fax please use black ink, a good pen and large letters easily to read. Then call your local dealer or Dataman's customer support center (phone number in the control program, menu **Help / About**). Please keep your manual, the programmer and the completed "**DEVICE PROBLEM REPORT**" form (just

faxed) available, so that you can respond quickly to our questions.

- If your programmer is diagnosed as defective, consult your local dealer or Dataman about the pertinent repair center in your country. Please carefully include the following items in the package:
  - defective product
  - completed "**DEVICE PROBLEM REPORT**" form
  - photocopy of a dated proof of purchase

> ***Without all these items we cannot admit your programmer to repair.***

Note:

*You may find the "**DEVICE PROBLEM REPORT**" form:*

- *in **Appendix A** of this manual*
- *at our Internet site (**www.dataman.com**)*

If you have an unsupported target device

If you need to operate on a target device not supported by the control program for programmer, please do not despair and follow the next steps:

- Look in the device list of the latest version of the control program on our Internet site (section Device List, file corresponded to your programmer). Your new target device might already be included in this version! If yes, download the file PG4UWARC.exe and install the new version of the control program.
- Contact Dataman direct, filling up a "**Device Problem Report**" form following the instructions at the end of this form. We may need detailed data sheets of your target device and, if possible, samples. The samples will be returned to you after we include your target device in a new version of PG4UW.

Note:

See also AlgOR service in Appendix C in this manual.
*You may find the "**Device Problem Report**" form:*

- *in **Appendix A** of this manual*
- *at our Internet site (**www.dataman.com**)*

Warranty terms

The manufacturer, Dataman Programmers Ltd, gives a guarantee on failure-free operating of the programmer and all its parts, materials and workmanship for **three-year** (Dataman-48Pro and Dataman-40Pro) from the date of purchase. This warranty is limited to 25,000-cycles on DIL ZIF socket or 10,000-cycles on other ZIF sockets). If the product is diagnosed as defective, Dataman Programmers Ltd or the authorized repair center will repair or replace defective parts at no charge. Parts used for replacement and/or whole programmer are warranted only for the reminder of the original warranty period.

For repair within the warranty period, the customer must prove the date of purchase.

This warranty terms are valid for customers, who purchase a programmer directly from Dataman company. The warranty conditions of Dataman sellers may differ depending on the target country law system or Dataman seller's warranty policy.

The warranty does not apply to products that are of wear and tear or mechanically damaged. Equally, the warranty does not apply to products opened and/or repaired and/or altered by personnel not authorized by Dataman, or to products that have been misused, abused, accidentated or that were improperly installed.

For unwarrantable repairs you will be billed according to the costs of replacement materials, service time and freight. Dataman or its distributors will determine whether the defective product should be repaired or replaced and judge whether or not the warranty applies.

Please also see Troubleshooting section.

Manufacturer:
✉: Dataman Programmers Ltd
Station Road
Maiden Newton
Dorset
DT2 0AE
United Kingdom
☎: + 44 0 1300 320719
www.dataman.com, sales@dataman.com

Appendix

Appendix A - Device Problem Report form

*Visit please the **www.dataman.com** site and use the Problem Report form (Support section), if occurred any problem during work with programmable device and Dataman programmers. If you haven't access to Internet, please make a copy of this page to A4.*

DEVICE PROBLEM REPORT

Subject(title of problem):_____ Date:_____

Customer

Customer, name:_____ Distributor, name:_____
Address:_____ Date of purchasing:_____
Contact person and e-mail:_____ Date of sending registration card:_____

Information about product.

Programmer (type/modification):_____ Mains supply voltage:_____ V
Serial number: _____ Version of control program PG4UW:_____
Configuration (modules, converters): _____
Power supply unit: From delivery Other (output V and A): _____

Information about PC, to which is the programmer is attached.

Manufacturer/Type:_____ Desktop Notebook
Processor, speed:_____ LPT port location: motherboard ISA card PCI card
Operating system and version:_____ LPT port type: standard ECP/EPP 1284
Memory/free memory:_____ LPT port setting: SPP BIDIR EPP ECP

Information about device with which you have the problem.

Device type (full name, prefix/suffix including):_____ Package type: plastic ceramic ceramic/windowed
Vendor/logo:_____ All designation on the top _____
Package (DIL40, PLCC44, SOIC20, ...):_____ and on the bottom side of device_____

Precedence rating: in __ days in __ weeks in __ months
How often you work with this devices: still Y/N sometimes Y/N one-shot Y/N
Number of programmed device: approx. ___ pcs per year.
Samples are available? Yes (I'm sending it/attached) Yes No

Further questions.

- Did you have installed latest version of control program? Yes No
- Did you know thoroughly the features and correct behavior of programmer and programmed device? Yes No
- Is the socket of programmer or adapter free from dust and isn't out of life? Yes No
- Is the device with problem new or used? New Used
- Is the error reported for all of the tested devices? Yes No I have only one device
- Is the error reported for devices with other date code? Yes No I have only one batch
- During which procedure is an error reported? Read Program ID_check Insertion test
- Is the programmer successful in case of other types of devices? Yes No
- Does the error occur always or randomly? Always Randomly
- Does programmer work well with other PCs? Yes No Not tested
- What is the results of programmer selftest (if available)? OK Error

Please list the step-by-step description of all activities that invokes a problem. Please make your problem description as specific as possible - you can increase speed and chance to resolve a problem. Please mention any step that is known to cause the problem or any step that may prevent the problem. Please copy all error reports too - full content of LOG window is preferable. We recommended use command Diagnostics/ Create diagnostic report. Your comments and descriptions of expectations are welcomed. It's best, if you can send us the actual device with which the problem occurs. Use a separate sheet if necessary.

_____
_____
_____
_____

Note:

- *if you haven't installed the latest version of control program, you can get it from **www.dataman.com** page (Download section). It is very important to have latest version of software, because:*
  - *it is possible the problem you have is already solved by software update*
  - *we don't save older version of software. If ask you to "please perform next steps ...", your version of software may not behave in the same way as the latest one as used by us.*
- *in the case of sending samples, please attach to the package this declaration for customs: "Free sample(s), not for commercial sale. Value for customs purposes only: $10US"*

Why is it important to use the latest version of the control program?

- Semiconductor manufacturers continuously introduce new devices with new package types, manufactured by new technologies in order to support the need for flexibility, quality and speed in product design and manufacturing. To keep pace and to keep you up-to-date, we usually implement more than 500 new devices into the control program within a year.
- Furthermore, a typical programmable device undergoes several changes during its lifetime in an effort to maintain or to improve its technical characteristics and process yields. These changes often impact with the programming algorithms, which need to be upgraded (the programming algorithm is a set of instructions that tells the programmer how to program data into a particular target device). Using the newest algorithms in the programming process is the key to obtaining high quality results. In many cases, while the older algorithm will still program the device, they may not provide the level of data retention that would be possible with an optimal algorithm. Failure to not use the most current algorithm can decrease your programming yields (more improper programmed target devices), and may often increase programming times, or even affect the long term reliability of the programmed device.
- Occasionally, we make mistakes too…

Our commitment is to implement support for these new or modified parts before or as soon as possible after their release, so that you can be sure that you are using latest and/or optimal programming algorithms that were created for this new device.

Appendix B - AlgOR service

(Algorithms On Request)
AlgOR is a free service, by which we respond, as flexible as possible on the customer's request to implement programming support for new devices. This service may be used also for requesting new features of the control program.

AlgOR process is simple. The user sends to Dataman a request for additional support for XXX device to the control program (we may ask for up-to-date data sheets and samples, if needed). After completion, the user will obtain a new version of the control program with requested features. We will, of course, also return the borrowed samples. If we cannot satisfy your requirements (too expensive, algorithms not available, additionally module needed), we will promptly contact to you and propose an appropriate solution.

Note:

- *Please use "**AlgOR** (Algorithms On Request)" form and send it direct to Dataman.*
- *AlgOR service is **free** of charge. Therefore we do not accept any claims regarding this service. Dataman Programmers Ltd reserves the right to set the dispatching priority on the particular tasks according to its own judgment.*

*Visit please the **www.dataman.com** site and use the AlgOR form (Support section) to ask the new chip support. If you haven't access to Internet, please make a copy of this page to A4.*

AlgOR (Algorithms On Request) form

Subject (title of problem): _____
Date: _____

Customer, name: _____
Address: _____
_____
Contact person and E-mail: _____
Distributor, name: _____
Date of purchasing: _____
Date of sending registration card: _____

Programmer (type/modification): _____
Serial number: _____
Control program and version: _____

Information about device, you want to be supported
Device type (full name): _____
Vendor/logo: _____
Package (DIL40, PLCC44,...): _____

Precedence rating: *in __ days* *in __ weeks* *in __ months*
Device to be programmed: *still Y/N* *sometimes Y/N* *one-shot Y/N*
Number of programmed device: *approx. ___ pcs per year.*
Samples are available? *Yes* *Yes (I'm sending it/attached)* *No*

***Notes to request.** Description of requested change in control program.*
Enter please feature you want to the program will have.

_____
_____
_____
_____
_____

Notes.

- *look please at latest list of supported devices before you send this request to us.*
- *in case of sending samples, attach please to package next declaration for customs: "Free sample(s), not for commercial sale.*

Value for customs purposes only: $10US"

Appendix C - registration card

If the registration card is missing from your standard programmer delivery package please use a copy of the form below and send it to Dataman. We remind you that without the "Extended warranty" document only the standard 6-month warranty is granted.

✂----------

Programmer: _____
Serial number: _____
Name: _____
Company: _____
Department: _____
Address: _____
Post Code, City: _____
Country: _____
Phone/Fax/e-mail: _____
Date of purchase: _____
Purchase from: _____

Complete, please:

- what type of computer are you using:
  ❒Intel ❒AMD ❒Other
- operating system:
  ❒Windows 95/98/Me ❒Windows NT/2000/XP
- how did you first hear about Dataman ?
  ❒advert ❒dealer ❒other_____
- why did you choose this product?
  ❒price ❒quality ❒recommendation
  ❒features (please specify) _____
- comments:_____

_____
_____
_____
_____